

The best Linux guide money can buy!

LINUX MADE SIMPLE

The ultimate guide to open source computing



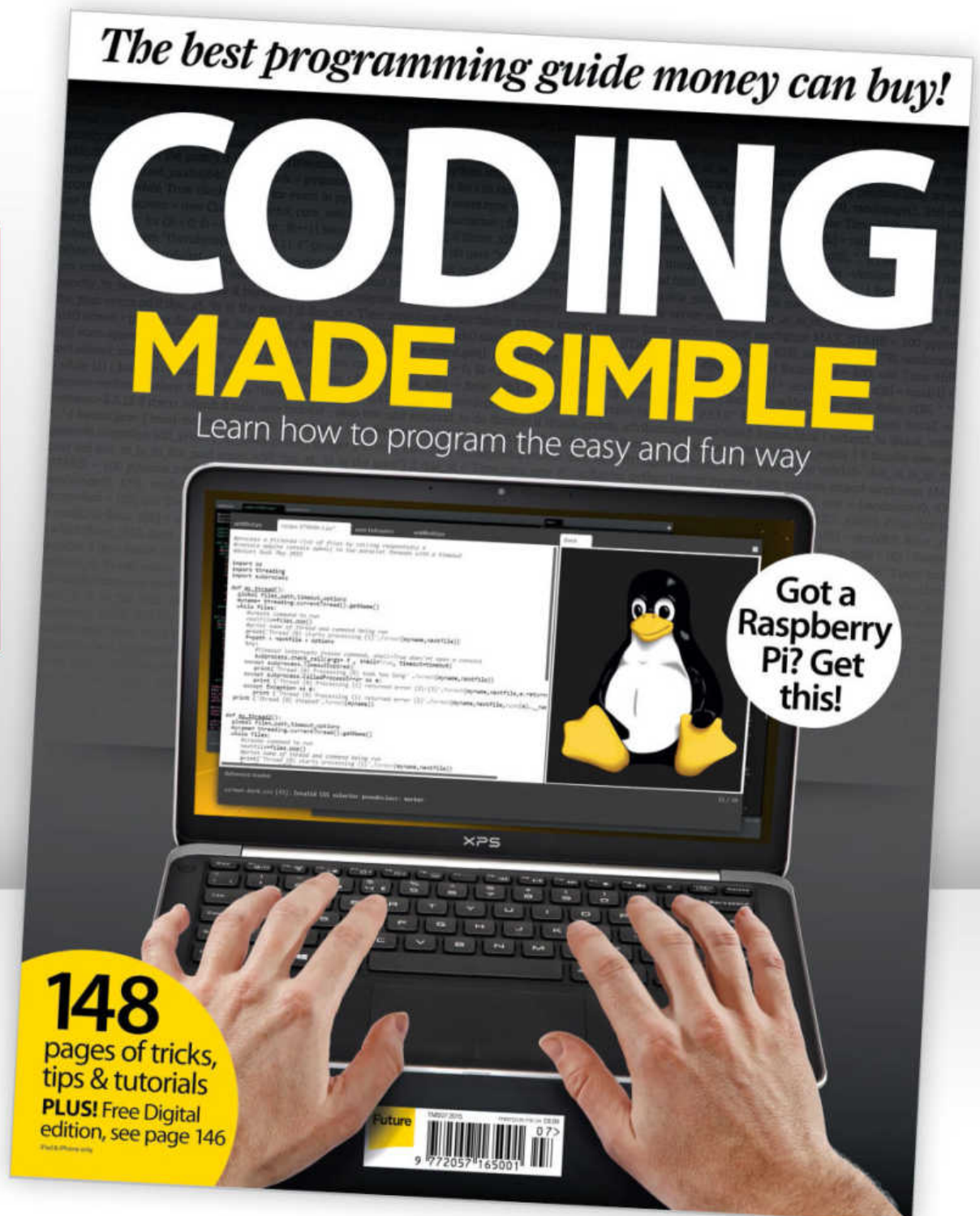
148
pages of advice
from the makers of
**LINUX
FORMAT**

Future

TMS09 2015

LEARN TO CODE WITH OUR GREAT TUTORIALS!

**OUT
NOW!**
WITH
FREE
DIGITAL
EDITION



DELIVERED DIRECT TO YOUR DOOR

Order online at www.myfavouritemagazines.co.uk
or find us in your nearest supermarket, newsagent or bookstore!

LINUX **MADE SIMPLE**

**THE ULTIMATE GUIDE TO
OPEN SOURCE COMPUTING**

LINUX

MADE SIMPLE

EDITORIAL TEAM

ART EDITORS
Fraser McDermott
Efrain Hernandez-Mendoza

EDITOR
Alex Cox

EDITOR-IN-CHIEF
Graham Barlow

CONTRIBUTORS
Neil Mohr, Mayank Sharma,
Joni Bidwell, Neil Bothwick,
Richard Smedley, Mihalios Tsoukalos,
Marco Fioretti, Matt Hanson

IMAGES
ThinkStock, Future Photo Studio

MANAGEMENT

CONTENT & MARKETING DIRECTOR
Nial Ferguson

HEAD OF CONTENT & MARKETING, TECH
Nick Merritt

GROUP EDITOR-IN-CHIEF
Paul Newman

GROUP ART DIRECTOR
Steve Gotobed

MARKETING

MARKETING MANAGER
Richard Stephens

PRINT & PRODUCTION

PRODUCTION MANAGER
Mark Constance

PRODUCTION CONTROLLER
Vivienne Turner

CIRCULATION

TRADE MARKETING MANAGER
Juliette Winyard
Phone +44(0)7551 150984

LICENSING

LICENSING & SYNDICATION DIRECTOR
Regina Erak
regina.erak@futurenet.com
Phone +44(0)1225 442244
Fax +44 (0)1225 732275

SUBSCRIPTIONS

UK reader order line & enquiries: 0844 848 2852
Overseas reader order line & enquiries: +44 (0)1604 251045
Online enquiries: www.myfavouritemagazines.co.uk

PRINTED IN THE UK BY

William Gibbons on behalf of Future.
Distributed in the UK by Seymour Distribution Ltd,
2 East Poultry Avenue, London EC1A 9PT. Phone: 020 7429 4000

Future Publishing Limited
Quay House, The Ambury, Bath, BA1 1UA, UK www.futureplc.com
www.myfavouritemagazines.co.uk
Phone +44 (0)1225 442244 Fax +44 (0)1225 732275

All contents copyright © 2015 Future Publishing Limited or published under licence. All rights reserved. No part of this magazine may be reproduced, stored, transmitted or used in any way without the prior written permission of the publisher.

Future Publishing Limited (company number 2008885) is registered in England and Wales. Registered office: Quay House, The Ambury, Bath, BA1 1UA. All information contained in this publication is for information only and is, as far as we are aware, correct at the time of going to press. Future cannot accept any responsibility for errors or inaccuracies in such information. You are advised to contact manufacturers and retailers directly with regard to the price and other details of products or services referred to in this publication. Apps and websites mentioned in this publication are not under our control. We are not responsible for their contents or any changes or updates to them.

If you submit unsolicited material to us, you automatically grant Future a licence to publish your submission in whole or in part in all editions of the magazine, including licensed editions worldwide and in any physical or digital format throughout the world. Any material you submit is sent at your risk and, although every care is taken, neither Future nor its employees, agents or subcontractors shall be liable for loss or damage.



Future is an award-winning international media group and leading digital business. We reach more than 49 million international consumers a month and create world-class content and advertising solutions for passionate consumers online, on tablet & smartphone and in print.

Future plc is a public company quoted on the London Stock Exchange (symbol: FUTR).
www.futureplc.com

Chief executive Zillah Byng-Maddick
Non-executive chairman Peter Allen
Chief financial officer Richard Haley

Tel +44 (0)207 042 4000 (London)
Tel +44 (0)1225 442 244 (Bath)

We encourage you to recycle this magazine, either through your usual household recyclable waste collection service or at recycling site.



When you have finished with this magazine please recycle it.



We are committed to using only magazine paper which is derived from well managed, certified forestry and chlorine-free manufacture. Future Publishing and its paper suppliers have been independently certified in accordance with the rules of the FSC (Forest Stewardship Council).

Welcome!

Linux might seem intimidating, but you've come to the right place if you want to get started.



Every time I come to write an introduction to one of these guides, I seem to relate a story of how I've ruined something by being unprepared. This guide is no different. My first

experience with Linux was one of desperation: I really *had* to try it, being an unabashed computer nerd, and when I finally got my hands on an early copy of Mandrake Linux – in a box, on a real CD-ROM! – I set to installing it without thinking. I didn't need that software or those documents, it seems. I also didn't need a working computer, clearly, as Linux wasn't particularly friendly when it came to compatibility at the time.

Things have changed. Modern Linux distributions tend to work first time with

whatever hardware you have. Every recent distro I've tried installs nicely alongside other operating systems like Windows and other flavours of Linux. You can also get by – just – without ever touching the command line, as long as you're not looking to push your OS too far.

However you want to use Linux, from GUI to command line and beyond to more practical purposes, Linux Made Simple has you covered. I've collected together a host of guides and tutorials from my expert colleagues at Linux Format magazine, which will get you up and running and take you further, too. And if you get hooked, I'd highly recommend subscribing to LXF (see page 144). You won't find a better Linux magazine.

Alex Cox, Editor

The **MADE SIMPLE** Manifesto

Made Simple books are designed to get you up and running quickly with a new piece of hardware or software. We won't bombard you with jargon or gloss over basic principles, but we will...

✓ Explain everything in plain-English so you can tackle your new device or software with confidence and really get to know how to use it

✓ Break instructions down into easy-to-follow steps so you won't be left scratching your head over what to do next

✓ Help you discover exciting new things to do and try – exploring new technology should be fun and our guides are designed to make the learning journey as enjoyable as possible for you

✓ Teach you new skills you can take with you through your life and apply at home or even in the workplace

✓ Make it easy for you to take our advice everywhere with you by giving you a free digital edition of this book you can download and take with you on your tablet, smartphone or laptop – see page 146 for more details on this offer

How are we doing? Email techbookseditor@futurenet.com and let us know if we've lived up to our promises!

LINUX

MADE SIMPLE

Learn how to upgrade, customise and master the free OS with a collection of tutorials and guides from the Linux experts!

Get Started

Baffled? Too scared to even start? Learn the basics from the ground up

- 10** What is Linux?
- 21** Get into Linux
- 31** Escape Windows
- 40** Switch from XP

Hardware

Linux is still best if you're using hardware that suits it; here's what to use and how

- 47** Build a Linux PC
- 56** Install on a Chromebook
- 60** Build a Steam machine
- 64** Raspberry Pi 2 hands on

Software

Improve your experience with top distributions and applications

- 73** Pick the right distro
- 84** Low resource distros
- 90** Top 100 Linux tools
- 98** Low resource applications

In-Depth

Get ready to learn some serious skills that will make you a master of Linux

- 104** Terminal basics
- 106** Apt-get in action
- 108** Disk management
- 110** Archiving
- 112** Core terminal programs
- 114** Terminal tricks
- 116** User accounts
- 118** Built-in help
- 120** Sysadmin basics
- 124** Super-user control

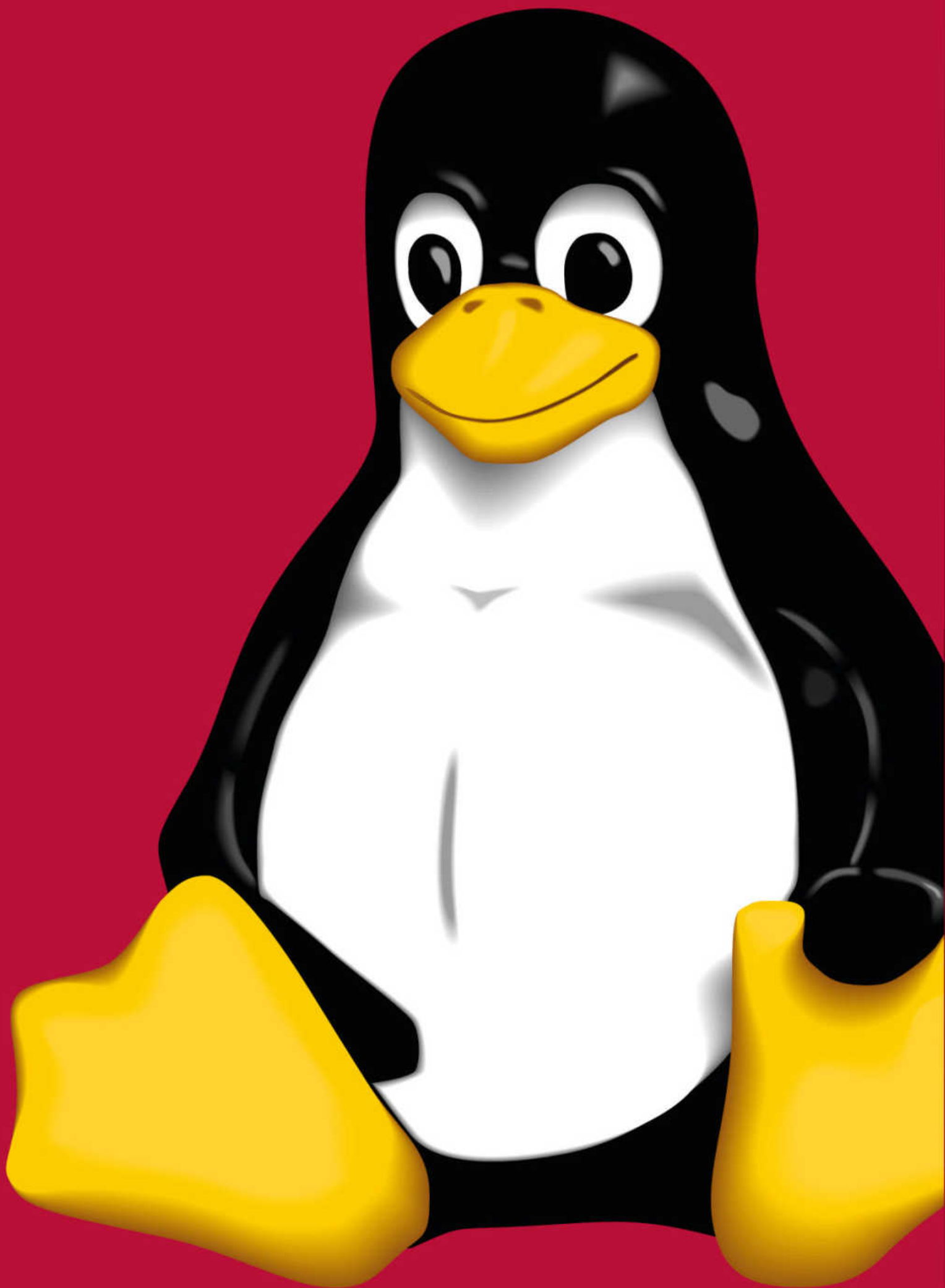
Projects

Do something perfectly practical with your new Linux knowledge

- 128** Build your own router
- 132** Try a microkernel OS
- 136** Repair broken systems
- 140** Analyse network traffic

Get Started

What is Linux?.....	10
Get into Linux.....	21
Escape Windows.....	31
Switch from XP.....	40



What is Linux?

There's... a penguin? And something called a distro? Don't worry: we've got the answers.

The word 'Linux' is one of the most used in this book, but what does it mean? It means different things to different people, from the purist who considers it to be the kernel, to the GNU advocate who sees it as a part of GNU/Linux and the new user who thinks it is another name for Ubuntu. In truth, Linux is all of these, depending on your point of view. Strictly speaking, the term Linux used alone refers to the kernel of the operating system, while GNU/Linux is the whole operating system, comprising the Linux kernel and GNU tools – either would be useless without the other (or one of its alternatives).

If you then add a collection of application software, along with some tools to manage the whole thing, you have a distro, such as Ubuntu.

There are lots of individual components that make up the operating system we know as Linux, but they are not that individual – they all have to fit together, so here we will try to explain how the whole is the sum of its parts, and what those parts do.



What is an OS? What is a distro?

An operating system can be defined as the software needed to enable the applications to run on the hardware – as such, it consists of several interleaved layers. At the heart we have the kernel, which interacts directly with the hardware through its drivers and allows other software to use that hardware. On top of that, we have various layers that handle things such as input devices, networking, sound and

video. Normally, you don't need to know anything about this. It can be helpful to know some of it when things go wrong, but even then it is not essential, especially if you can find someone to fix the computer for you. But if you are reading this book, there is a good chance that you are interested in what is happening 'down below', so we will try to give you an idea of what goes where, and what it does when it gets there. A Linux

distribution is just that, a way of distributing a Linux-based operating system and accompanying software. At the start, it was just the files the OS needs and a way of installing them on your computer. Along the way, distros acquired package managers, update tools, configuration GUIs and a host of other niceties. However, underneath the user-friendly (or not if you are a Gentoo user) gloss, all distros are still Linux.

The kernel

The nerve centre at the heart of your Linux operating system.

The kernel is the beating heart of the system, but what is it? The kernel is the software interface to the computer's hardware. It communicates with the CPU, memory and other devices on behalf of any software running on the computer. As such, it is the lowest-level component in the software stack, and the most important. If the kernel has a problem, every piece of software running on the computer shares in that problem.

The Linux kernel is a monolithic kernel – all the main OS services run in the kernel. The alternative is a microkernel, where most of the work is done by external processes, with the kernel doing little more than co-ordinating.

While a pure monolithic kernel worked well in the early days, when users compiled a kernel for their hardware, there are so many combinations of hardware nowadays that building them all into the kernel would result in a huge file. So the Linux kernel is now modular, the core functions are in the kernel file (you can see this in `/boot` as `vmlinuz-version`)

while the optional drivers are built as separate modules in `/lib/modules` (the `.ko` files in this directory).

For example, Ubuntu 14.04's 64-bit kernel is 5MB in size, while there are a further

3,700 modules occupying over 100MB. Only a fraction of these are needed on any particular machine, so it would be insane to try to load them all with the main kernel. Instead, the kernel detects the hardware in use and loads the relevant modules, which become part of the kernel in memory, so it is still monolithic when loaded even when spread across thousands of files. This enables a system to react to changes in hardware. Plug in a USB memory stick and the **usb-storage** module is loaded, along with the filesystem module needed

to mount it. In a similar way, connect a 3G dongle, and the serial modem drivers are loaded. This is why it is rarely necessary to install new drivers when adding hardware; they're all there just waiting for you to buy some new toys to plug in. Computers that are run on specific and unchanging hardware, such as servers, usually have a kernel with all the required drivers compiled in and module loading disabled, which adds a small amount of security.

If you are compiling your own kernel, a good rule of thumb is to build in drivers for hardware that is always in use, such as your network interface and hard disk filesystems, and build modules for everything else.

Even more modules

The huge number of modules, most of which are hardware drivers, is one of the strengths of Linux in recent years – so much hardware is supported by default, with no need to download and install drivers from anywhere else. There is still

some hardware not covered by in-kernel modules, usually because the code is too new or its licence prevents it being included with the kernel (yes ZFS, we're looking at you). The drivers for Nvidia cards are the

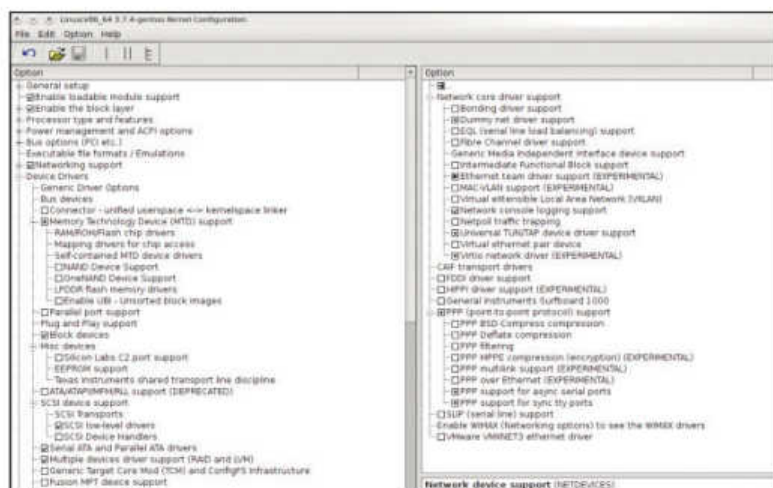
best known examples. Usually known as third-party modules, although Ubuntu also refers to 'restricted drivers', these are installed from your package manager if your distro supports them. Otherwise, they have to be compiled from source, which has to be done again each time you update your kernel, because they are tied to the kernel for which they were built.

There have been some efforts to provide a level of automation to this, notably DKMS (Dynamic Kernel Module Support), which automatically recompiles all third-party

modules when a new kernel is installed, making the process of upgrading a kernel almost as seamless as upgrading user applications.

Phrases that you will see bandied about when referring to kernels are "kernel space" and "user space". Kernel space is memory that can only be accessed by the kernel; no user programs (which means anything but the kernel and its modules) can write here, so a wayward program cannot corrupt the kernel's operations. User space, on the other hand, can be accessed by any program with the appropriate privileges. This contributes towards the stability and security of Linux, because no program, even running as root, can directly undermine the kernel.

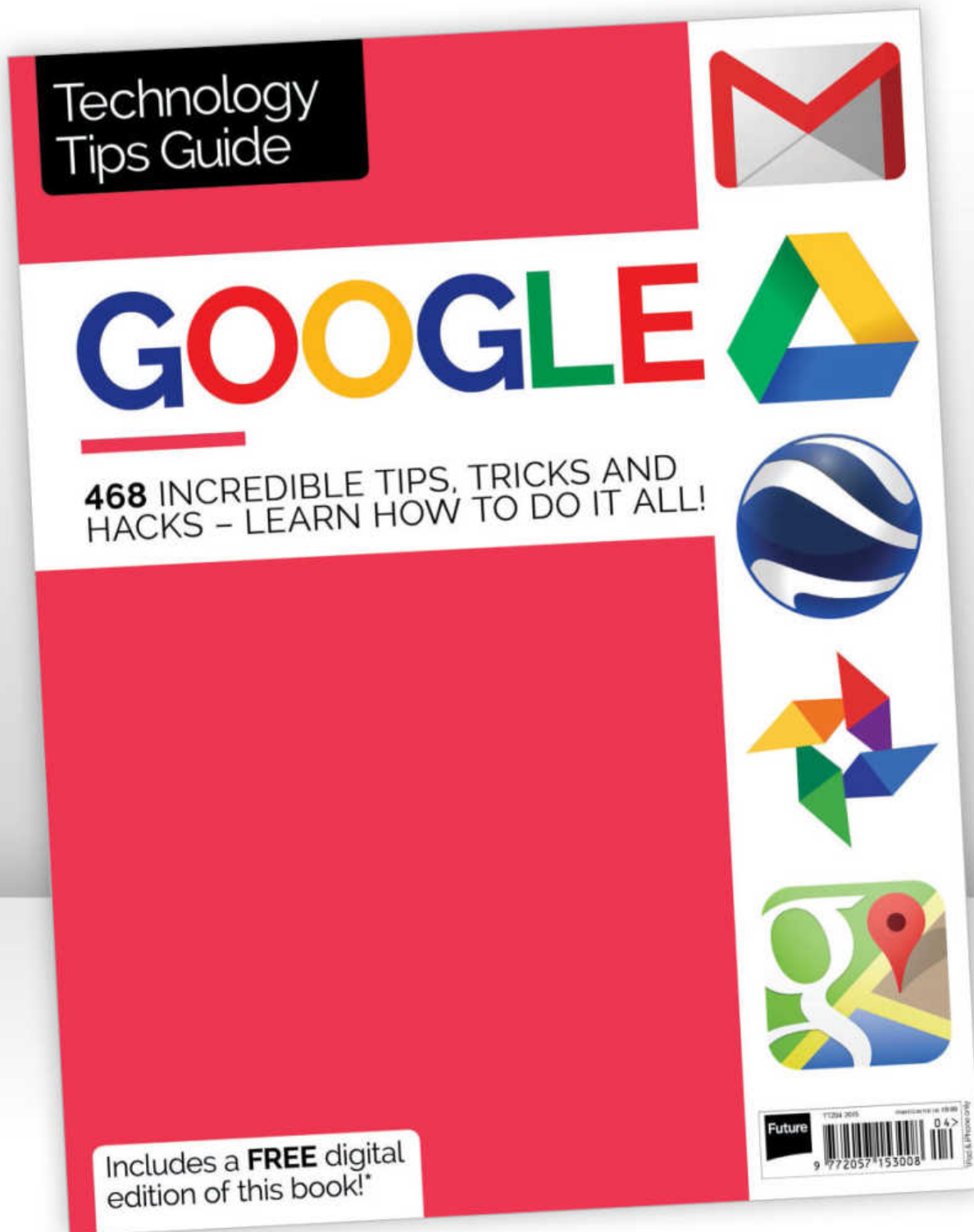
“So much hardware is supported, with no need to download drivers”



» The number of options when building a kernel is truly staggering, and the majority of them relate to hardware device support. Aren't you glad we have distro maintainers to work it all out for us?



GET THE MOST FROM GOOGLE



**OUT
NOW!**
**WITH
FREE
DIGITAL
EDITION**



DELIVERED DIRECT TO YOUR DOOR

Order online at www.myfavouritemagazines.co.uk
or find us in your nearest supermarket, newsagent or bookstore!



T3

**HELPING YOU
MAKE THE MOST
OF LIFE IN TODAY'S
CONNECTED WORLD**



ONLINE • PRINT • TABLET

**AUDIO TECHNICA
ATH-MSR7**
Escape to a world of
high resolution audio

SAMSUNG JS9000
Relax with the latest
home entertainment

CARL ZEISS VR
Embrace a new era of
virtual reality gaming

SONY XPERIA Z3
Putting you in control
of your life and home

APPLE WATCH
Your health and
fitness upgraded

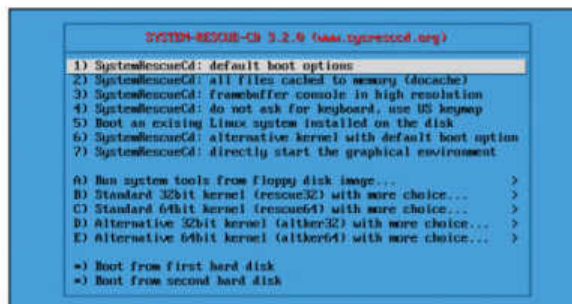
LIFE'S BETTER WITH T3

t3.com



notice any difference. There are also moves to replace the traditional SysVinit system, which has been around for many years. Ubuntu introduced *Upstart*, while Red Hat and Fedora are championing *systemd*.

While they all do the same basic job – start the tasks that need to be started to get the OS running – the methods differ. The main difference is that SysVinit is sequential – each service must be started before the next is tackled; one slow starting service affects everything else. *Upstart* and *systemd* start services in parallel, avoiding such bottlenecks. Of course, there are those who argue that Linux is so stable that boot times are irrelevant – if you hibernate instead of shutting the machine down, reboots become rare.



» **Grub** is the most popular bootloader. Live CDs and USB sticks are more likely to use *isolinux* or *syslinux*.



Libraries

The logic behind sharing functions between programs.

Linux uses libraries to share code between applications. If a program, *foo*, uses functions that could be useful elsewhere, it places them in *libfoo*. Then when another program, such as the imaginatively named *bar*, wants to use the same function, it has only to link to *libfoo* rather than reinventing the wheel.

This means that there is only one copy of the code on your computer; if either project discovers a bug in the code, it will be fixed for both. It also introduces the concept of dependencies; both *foo* and *bar* depend on *libfoo* and are useless without it. This led to the phenomenon of ‘dependency hell’, where trying to install a program errored out with a list of unsatisfied dependencies, and trying to install those gave more dependencies. This is largely an unpleasant memory nowadays, as distro repositories became more comprehensive and package managers better at sorting things out.

If you stick with your distro’s package manager and repositories, all dependencies should be taken care of without you even having to think about them. Try installing **somerandom.deb** or **somerandom.rpm** you downloaded from **www.somerandomsite.com** and you’ll soon discover

“You can see which libraries a program is linked to with ldd”

why you should let the package manager take care of things. One solution proposed to this is that all programs should be compiled statically. This means that instead of dynamically linking to the code in *libfoo* and loading when needed at run time, *foo* and *bar* each include the code in their executable programs. This means each program file is a standalone object with no dependencies; it can also make it a lot larger than it would be with dynamic linking and means that if a bug or security flaw is found in the *libfoo* code, both *foo* and *bar* will need to be recompiled and repackaged for your distro to fix the situation. Generally,

dynamic linking is preferred on non-embedded devices, but there is one place where statically linked programs are useful: in an *initramfs* loaded at boot time, because it avoids the

need to include libraries in the ramdisk image. If you are curious, you can see which libraries any program is linked to with the **ldd** command.

ldd /usr/bin/someprogram shows all the libraries that program needs, and the libraries they need and so on, until you almost always end up at **libc** – the granddaddy of Linux libraries.

Package managers

The great flexibility of Linux distributions means that most elements can be changed. Default applications, desktops, even kernels can be swapped around, so it’s best to think of a Linux distribution such as Fedora or Ubuntu as merely a starting point for any customisation that you want to do.

The one thing that can’t be changed so easily is the package manager, so the only way to try a different package manager is to try a different distro. Try comparing SUSE’s *Yast* with Debian’s *Synaptic*, for example, and you’ll be amazed at the difference that such a fundamental tool can make to your experience of using Linux.

```

$ ldd /usr/bin/k3b
linux-vdso.so.1 (0x00007fffef61ff00)
libk3bdevice.so.6 => /usr/lib64/libk3bdevice.so.6 (0x00007ff9cbec000)
libk3blib.so.6 => /usr/lib64/libk3blib.so.6 (0x00007ff9c811000)
libk3ddb.so.4 => /usr/lib64/libk3ddb.so.4 (0x00007ff9c3c000)
libk3file.so.4 => /usr/lib64/libk3file.so.4 (0x00007ff9c31000)
libk3io.so.5 => /usr/lib64/libk3io.so.5 (0x00007ff9c0e000)
libk3notifyconfig.so.4 => /usr/lib64/libk3notifyconfig.so.4 (0x00007ff9c0c370)
libk3support.so.4 => /usr/lib64/libk3support.so.4 (0x00007ff9b932000)
libsolid.so.4 => /usr/lib64/libsolid.so.4 (0x00007ff9b636000)
libQtWebkit.so.4 => /usr/lib64/qt4/libQtWebkit.so.4 (0x00007ff99b7c000)
libkcmutils.so.4 => /usr/lib64/libkcmutils.so.4 (0x00007ff9993000)
libQt3Support.so.4 => /usr/lib64/qt4/libQt3Support.so.4 (0x00007ff99455000)
libQtXml.so.4 => /usr/lib64/qt4/libQtXml.so.4 (0x00007ff99210000)
libkdeui.so.5 => /usr/lib64/libkdeui.so.5 (0x00007ff98b73000)
libkdecore.so.5 => /usr/lib64/libkdecore.so.5 (0x00007ff980e2000)
libQtCore.so.4 => /usr/lib64/qt4/libQtCore.so.4 (0x00007ff983a4000)
libQtGui.so.4 => /usr/lib64/qt4/libQtGui.so.4 (0x00007ff974f3000)
libstdc++.so.6 => /usr/lib/gcc/x86_64-pc-linux-gnu/4.6.3/libstdc++.so.6 (0x00007ff96e45000)
libc.so.6 => /lib64/libc.so.6 (0x00007ff96e45000)
libpthread.so.0 => /lib64/libpthread.so.0 (0x00007ff96c20000)
libm.so.6 => /lib64/libm.so.6 (0x00007ff96930000)

```

» Shared libraries enable a more efficient system, by sharing code between applications. Here are just some of the libraries the *K3b* disc burner links to.

»

Graphics

How your Linux box stays looking so tickety-boo.

The *X Window System* is the standard basis for providing a graphical interface. While the likes of KDE and Gnome provide the user interface and eye candy, it is through *X* that they communicate with the hardware. For many years, this was a mass of arcane configuration options and required a lengthy configuration file containing things such as modelines that specified the likes of pixel clock rates and sync frequencies.

These days, most systems will run without any configuration file at all. Massive improvements in hardware detection mean that a system with a single video card and single display will 'just work'. You may need to install extra drivers to get 3D acceleration if you are using, for example, an Nvidia card, but otherwise you just boot the computer and start clicking your mouse. *X* has a client/server

architecture. *X* itself runs as the server, maintaining the display; client programs then communicate with the server, telling it what to draw and where.

Legacy features

This may seem excessively complex, but it uses local sockets to communicate between the clients and server, so there is no significant performance hit. One clear advantage of this method is that the client and server do not have to be running on the same computer. You can connect to another computer by SSH and, providing the configuration gives permission for this, run a program on the remote computer and have its GUI

displayed on your local screen.

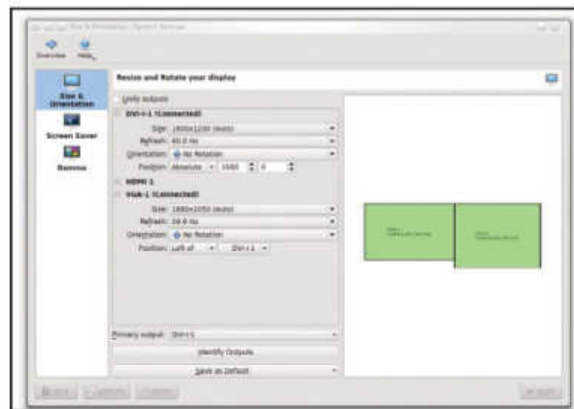
This is different from the likes of VNC because only the one application's window is displayed locally, and it only appears locally – not on the remote computer. A VNC

connection mirrors the whole desktop on both computers.

Some consider the client/server architecture to be overly complex, so there are moves to develop more simple methods of running a graphical display. The most advanced is *Wayland*. This takes a different approach; not only is the old client/server setup gone, but *Wayland* leaves the rendering of windows and other display elements to the client applications, usually using *OpenGL* and *Cairo*. This simplifies *Wayland*; *X* contains a lot of legacy rendering code that's required by the *X* specification but never used. By giving control to the clients, *Wayland* can be lighter, more efficient and future-proof. It also means your graphical software has more control over how the GUI is displayed.

“These days, most systems will run without any configuration file”

Tools such as KDE's monitor settings help with things like setting up dual monitors, but for a single display you shouldn't need to configure *X* at all.



Daemons

If you ever disable the splash screen most distros use to cover the boot messages, you will see a screen full of information about services being started. What are these services, and are they all necessary? The services are the programs that run in the background, making the computer as useful as it is. Some deal with networking, others handle hardware detection and configuration, while more are the traditional software services, or daemons, that provide functions to other programs when needed.

The answer to the second part of that question is most likely to be “no”. While some of these services are used by almost all systems, such as the **syslog** daemon that handles writing information to system log files, others may not be needed. There is no need to start CUPS, the printing system, if you don't have a printer available. Similarly, the *MySQL* database server may not be needed, nor the SSH daemon if you

have only one computer on your network. So spending half an hour experimenting could shave a second off your boot time.

You may also save some resources by not starting unnecessary services, but once loaded these daemons consume almost no system resources, and even the memory that they use can be swapped out if they are not called. So only disable those services you know you will never need. Having them patiently listening on a network port or socket makes the operation of your client programs that bit more efficient. Programs don't need to include, or load, code for opening, writing to and closing log files, they just call the **syslog()** function with the log text, and the daemon takes care of the rest. **Syslog** is an important service – when something goes wrong, this is often the first place to look, as most programs send error messages to the system log (usually at **/var/log/messages**).



It is possible to reduce your boot time by only running the services you need.

Why are background services called daemons? There are a few explanations; we prefer the story that daemons were beings in Greek mythology that handled tasks that the gods could not be bothered with.

Networking

How your computer talks to others.

Networking is core to Linux. Even on a standalone machine with no connection to a local network, let alone the internet, networking is still used. Many services run on a client/server model, where the server runs in the background waiting for instructions from other programs. Even something as basic as the system logger runs as a networked service, allowing other programs to write to log files. The X graphics system is also networked, with the X server running the desktop and programs, telling it what they want displayed. This is why it is so simple to run X programs on a remote desktop – as far as the system is concerned, there is no major difference between that and opening a window locally.

Running `ifconfig` will always show at least one interface, called `lo` with an address of `127.0.0.1` – this is used by the computer for talking to itself, which is regarded as a more sane activity for computers than people. Most other networking is based on TCP/IP, either over wired Ethernet or wireless, but there are other types of network in use. All distros and desktops include good tools for configuring and maintaining TCP/IP networks, from the fairly ubiquitous *NetworkManager* to individual tools such as Gnome's network configuration tool or OpenSUSE's *Yast*. More recent

additions to the networking scene include 3G mobile communications and PAN (Personal Area Network) technologies such as Bluetooth. Using a 3G mobile broadband dongle is usually simple, either using *NetworkManager* or your desktop's PPP software. Yes, 3G modems really do work like modems using dialscripts and everything, but without the cacophony of squawks as you connect (younger readers should ignore the last statement). Most problems with 3G are caused by trying to set them up in a poor signal area rather than with either the hardware or software support in Linux.



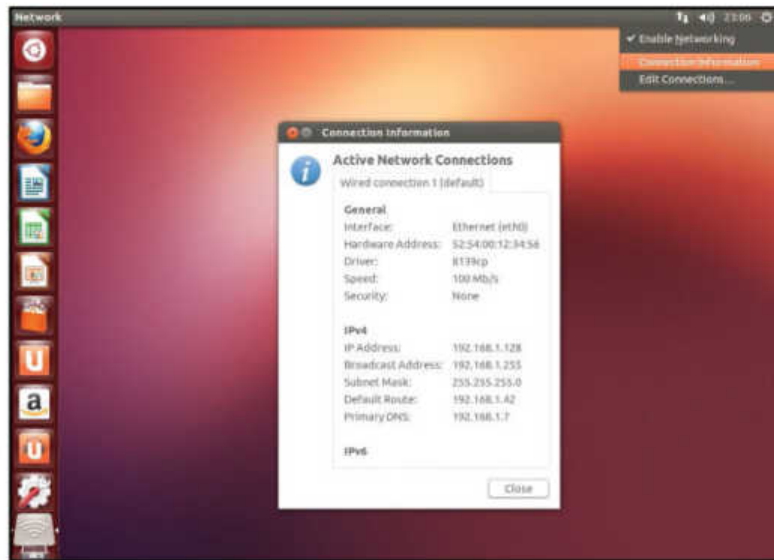
The protocol of kings

Bluetooth is becoming more important as mobile devices proliferate, and the number of input and output devices using it is increasing. It's not only phone and tablet users who benefit – a Bluetooth mouse and speakers can enhance the use of a laptop when at your desk, without having to plug everything in before you can start working. PulseAudio (see the section on sound) makes this easier, because it can switch between devices when they are detected.

Storage

Storing data on a hard disk can involve several layers in itself. All physical storage (as opposed to network storage) in Linux revolves around block devices, so called because disks store data in blocks. A block device like `/dev/sda1` does indeed refer to blocks, physical areas on the disk, and a collection of them as a disk partition. On top of that we have a filesystem, which is how the data is stored in a sensible structure of directories and files, containing both data and metadata.

What's the difference? Let's say you save a file containing some text. The data in that file is the text, but the file has other attributes: there is the owner of the file, the time they created it, the time they last modified it, the time it was last read and who has permission to read or modify it. This is the information you see when you `ls -l` a file, or inspect its properties in your file manager, and this is stored by the filesystem. The standard filesystem in use nowadays is `ext4`, but there are alternatives such as `ext3`, `ReiserFS`, `XFS`, `JFS` and, of course, `FAT` and `NTFS` from the world of Windows.



› Networking is core to the operation of a Linux system. The localhost interface is set up automatically; for the rest we have programs such as *NetworkManager*.

Other Linuxes

Everything we have covered relates to Linux running on desktops, laptops and servers – traditional computer hardware if you like, but there are other environments where Linux is used. Many embedded devices, from routers to PVRs and set-top boxes, run Linux, and in many ways it's similar to the Linux we know and love. If

your router allows SSH access, you will often feel at home as soon as you log in.

There is another class of device that has seen a huge uptake in recent years and runs a rather different Linux. No prizes for guessing we are referring to the smartphone and its tablet siblings, running Android. Android is Linux, it uses

a Linux kernel, but it is not GNU/Linux. The kernel may be based on the same source code, but everything running on top is different. The principles are similar in some ways, but the implementation is very different – although you will find familiar command line tools if you can get to a shell prompt on your phone.



Desktops

Gnome, KDE Cinnamon, Unity – we'll just call it the user interface.

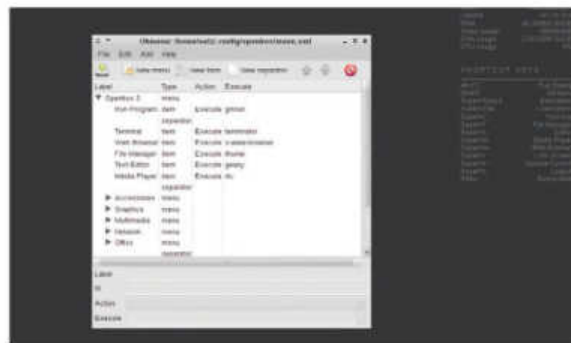
If you consider the kernel to be the lowest level of the system, the highest level is the user interface. Everything else, from the kernel through the drivers and hardware interfaces, is of no use until you can use the computer. This generally means a graphical desktop, and here we come across more layers. X (or maybe Wayland in the future) simply provides a blank canvas. You then need something to provide the niceties of a windowed interface, and that something is the window manager.

In the past, window managers were standalone systems, and there are still plenty of these available, such as *OpenBox* or *Enlightenment*, but nowadays they are often part of a larger desktop environment. Strictly speaking, a window manager is responsible for the handling of windows on the desktop, their opening, closing, placement and other manipulations. Over time, they grew to incorporate other features, such as taskbars and program launcher menus, until they developed into desktop environments.

Software collections

A desktop environment is simply a more or less integrated collection of utilities to provide the features needed to run a complete desktop. Running programs, manipulating their windows, keeping track of what is going on and enabling programs to communicate with one another are all features of desktop environments, but they still have a window manager at their heart – *KWin* for KDE and *Metacity* in Gnome to name but two.

What sets a desktop environment apart from a window manager is the level of integration. This is



There are also plenty of lightweight window managers, like *OpenBox* running on *CrunchBang* here.

particularly evident in KDE, where everything works around a common core, and programs not only communicate with one another, but an instance of one program can even be embedded in the window of another.

While it may not make much sense to use *KWin* on Gnome, you may want to try one of the more specialist window managers that offer greater control over window handling, or use a different method of displaying them. There are tiling window managers, like *awesome* and *xmonad*, that resize windows so they all fit on the desktop (KDE has its own option to behave like this). There are also window managers designed to be controllable with the keyboard, and minimal window managers that are useful for specialist systems that run a single program in a full-screen window and don't want any widgets cluttering up the place.



Gnome, KDE, Unity, Cinnamon, Mate – we aren't exactly short of choice when it comes to desktop environments, but how many of you have tried more than a couple?



Sound

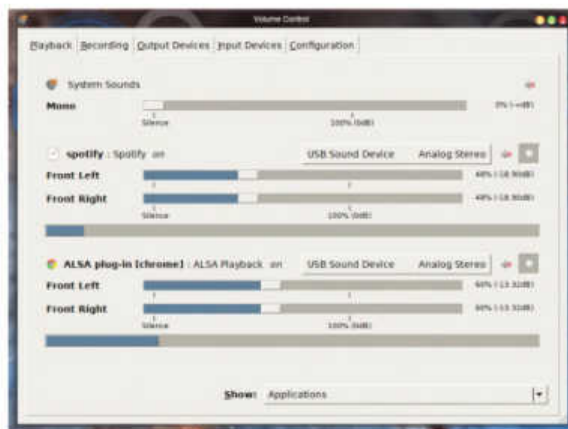
A once thorny subject.

The Linux sound system has been through many changes. We had OSS (Open Sound System) for years, before ALSA (Advanced Linux Sound Architecture) came along. ALSA is still with us, but we also have PulseAudio and Jack now.

ALSA supports multiple sound cards, each of which may have multiple inputs and outputs. It also provides support for hardware mixing, where the hardware supports it, and software mixing where it does not. This removes the need for sound managers, as provided by KDE and Gnome in the past, to enable more than one application to play sound at the same time. ALSA works at a low level, close to the hardware, so it gives low latency. Most hardware is directly supported now, and installing a distro should result in sound working from the first boot. ALSA is a combination of kernel code and user-space applications. It also provides an API so that other programs can control it directly, like the mixer control panels included with desktop environments.

PulseAudio performance

PulseAudio is a newer audio framework, but it is not a replacement for ALSA. Instead, it sits on top of the kernel audio system, providing a greater amount of control. It works as a server, accepting input from sources and forwarding it to sinks (output hardware or capture software). In many cases, the sink is ALSA, and the source can be an ALSA driver, too, for applications that don't directly support PulseAudio. Hence



➤ If anyone tries to tell you that PulseAudio is complicated, it's best not to argue with them. Not that the complexity of this layout matters too much if it just works for you.

you can end up with an application sending output to an ALSA device, which intercepts the stream and routes it through PulseAudio back to ALSA. It is no surprise that many found PulseAudio complex. A good setup should render all of this chicanery transparent to the user, which is where we are now with distro installers and PulseAudio, so most of the time we are back at the 'just works' situation of ALSA, but with better support for multiple devices. ALSA supports multiple output devices, but the default is a global setting. PulseAudio allows you to direct music through speakers while using a Bluetooth headset for VOIP calls. It also allows for less complex but equally useful separation, such as separate volume settings for each application. PulseAudio is network-aware – it can be used to find other PulseAudio servers and play audio through their speakers – great for streaming music around the house.

JACK (Jack Audio Connection Kit) is a sound server designed for professional audio applications. Its forte is providing low-latency real-time connections between applications, for audio and MIDI data. It is not needed for typical desktop use, only for budding musicians.

Printing

CUPS and drivers.

While open source encourages choice, and therefore several programs that do the same thing but in slightly different ways, there are some areas where one program is virtually unchallenged. **X.org** is used universally to provide graphical displays, and **CUPS** maintains a similar position in the printing arena.

If you have a printer attached to your Linux box, you need two things – CUPS and drivers for your printer. In many cases, these come together. CUPS is a server that sits in the background waiting for print requests. Just about any program that prints knows about the Internet Printing Protocol (IPP) that CUPS speaks. Hit 'Print' in your word processor or browser, and a window pops up showing your printer, and giving a choice if you have more than one.

The application only needs to send the data to be printed, usually as PostScript, to CUPS, which takes care of everything else – including waiting for you to remember to switch the printer on. CUPS does not even need to be running on the same computer; it is a networked service and a printer on one computer should be available to everyone on the same



➤ Taking care of printers with CUPS is as easy as following a few links in a browser, thanks to its built-in web interface.

network. CUPS is useless without drivers that tell it how to speak to the printer. It includes a large number of drivers by default, and many more are available by installing the **gutenprint** driver package (so many that your distro may well have installed this by default). HP also provides a driver (and scanner) driver package called **hplip**, which you need if you use its products.

However, some printer companies insist on providing their own drivers instead of having them bundled with CUPS, usually for licensing reasons. In that case, you have the choice of trawling the printer manufacturer's website for a driver package suitable for your system and installing it separately. After that, the drivers should appear in CUPS and your distro's printer configuration tool. The other choice is to check with **linuxprinting.org** before buying a printer, and stick to the more enlightened manufacturers.

techradar.pro

IT INSIGHTS FOR BUSINESS



THE ULTIMATE DESTINATION FOR BUSINESS TECHNOLOGY ADVICE

- Up-to-the-minute tech business news
- In-depth hardware and software reviews
- Analysis of the key issues affecting your business

www.techradarpro.com

twitter.com/techradarpro

facebook.com/techradar



Get into Linux

Curious about Linux but not sure how to traverse this unfamiliar territory? We'll help you get started so you can join the growing ranks of Linux users and get a better PC to boot.

There's never been a better time to get into Linux. It's slicker than ever, easier to install than ever and all the big-name distros like Ubuntu have just received updates.

One of the biggest impediments to

ships with a ton of software and you can download more with a single click.

Unlike proprietary OSes, a Linux flavour (or distro) is very malleable. You can swap out default apps or even its entire interface and replace it with something you choose. Choice is

another hallmark of Linux with multiple options from simple components to complex suites.

“To top it all, you can do everything you can on a Windows box.”

widespread Linux adoption is that you don't get Linux on a PC from Currys. But the advantages it offers over other mainstream OSes are well worth this extra step. For starters, Linux is open source, which is to say that you can legally download a copy of Linux and install it on all your computers. It also

Furthermore, besides being compatible with all your newer hardware, Linux can also turbocharge hardware that's past its glory days. To top it all, you can do everything you can on a Windows box. From streaming video to playing the latest games, Linux will work just as well as any other system.



Ubuntu

The most commonly known Linux distribution (often abbreviated to distro), Ubuntu pays special attention to desktop usability. In the 10 years of its existence, Ubuntu has galvanised the development of Linux on the desktop and is the go-to distro for third-party developers and vendors who want to run their wares on Linux.



Fedora

Red Hat's open source offering to the world, Fedora is known for adapting and offering new technologies and software to its users. Over the years, the distro has managed to find a clever balance between offering new features and stability to its users, which makes it popular with both new and experienced Linux campaigners.



Mageia

Although it's just had four releases to date, Mageia has a pedigree in usability that dates back to the 1990s. Mageia is a community project that's supported by a non-profit organisation, which is managed by a board of elected contributors. The distro is known for its customised user-friendly tools for managing the installation.

Get the UK's best-selling Linux magazine



DELIVERED DIRECT TO YOUR DOOR

Order online at www.myfavouritemagazines.co.uk
or find us in your nearest supermarket, newsagent or bookstore!

Install Linux

All you need to know to anchor Linux on your computer.

» **T**he Linux installation process is involved but it isn't actually that cumbersome. The exact installation steps are slightly different for every distribution, but in general the distro's graphical installer will guide you through the necessary steps pretty easily.

In essence, installing Linux is very similar to installing a piece of software, albeit with a few caveats:

» **Disk partitioning** Unlike a piece of software, installing Linux requires you to create a dedicated partition on your hard disk. This isn't an issue if Linux will be the only OS on this computer. However, if you're installing Linux alongside another OS, such as Windows, you'll have to take steps to preserve the existing data. Many Linux distros will offer to partition the disk for you automatically, though you can create partitions yourself with ease from within Windows using the Disk Management tool (see *Make Room for Linux*, below). The advantage of manually partitioning your disk is that you get to decide how much space to allocate to Linux.

When creating partitions remember to create two new partitions. The bigger one with at least 12GB of disk space is for the OS itself, which you'll format as ext4. You'll also need to create a second partition for what's called swap space. In simple terms, the swap partition extends the amount of physical RAM on your computer. A general rule of thumb for computers with a small amount of RAM (one or two gigabytes) is to create a swap partition that's twice as large as the amount of RAM on your computer. For computers with more RAM, it's best to create a swap partition that's the same size as the amount of RAM you have.

» **Securing data** During the installation process, many distros including Fedora and Ubuntu will give you an option to encrypt the Linux partition. This option gives you an added layer of security by insulating your data from unauthorised access. To enable this option you will need to supply a passphrase which will then act as the key to unlock the data.

Another important step during installation is setting up a root account. On most distros this step is part of the user creation process where you define the login credentials of your regular user account. The regular user doesn't have any permissions to modify the system while logging in as root gives you complete control over your system.

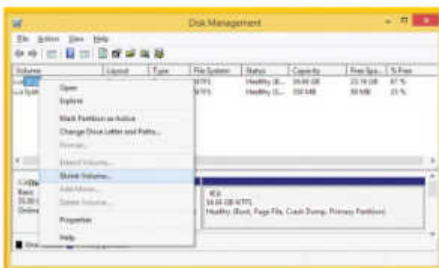
» **Dual boot** One software you should be familiar with when installing Linux is the bootloader. It's a small program that tells the computer where to find the different OSes on the disk. Most Linux distros use the *Grub 2* bootloader.

In general, you shouldn't have to do anything here, even when installing Linux on a Windows 8 computer that uses the UEFI BIOS with Secure Boot enabled. The latest versions of most mainstream distros, including Ubuntu and Fedora install a UEFI-compatible bootloader that will work correctly out of the box. However, since different vendors implemented UEFI differently, you might not get to the *Grub* bootloader screen and instead end up booting straight into Windows after installing Linux. In such a case, you should consider enabling the Legacy BIOS mode wherein the UEFI firmware functions as a standard BIOS. The option to enable Legacy BIOS is under the UEFI settings screen.

» **Testing before installation** Almost every mainstream distro, including Ubuntu, Fedora and Mageia allow you to boot into a 'live' environment, which lets you experience the distro without disturbing the contents of your hard disk. You can use the live environment to get familiar with the distro and also verify the compatibility of your hardware with the distro.

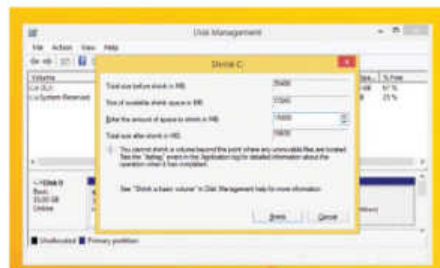
Also note that Linux distributions are distributed as ISO images. You can burn them to a CD or DVD, depending on their size, using the option to burn ISO images. You can also transfer ISO images to a USB drive. There are tools, such as *UNetbootin* and *Yumi* that will create bootable USB drives with the ISO of your distro, while Mageia recommends using the *Rufus* utility.

Make room for Linux: Resize a Windows partition



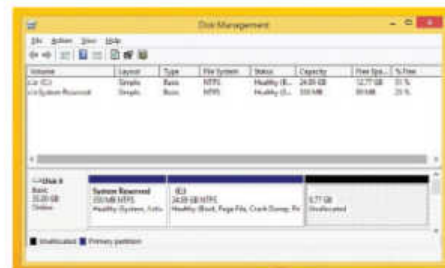
1 Shrink Windows

Before you can partition your disk you'll need to squeeze your Windows partition to free up some disk space for the new partition. Head to the Disk Management tool, and right-click your main partition that is typically assigned the drive letter C. Then select the Shrink Volume option from the pop-up menu.



2 Create new partition

The Shrink dialog box shows you the total size of the drive and the maximum amount of shrink space available. You cannot squeeze more space out of a drive than the size shown here. To create a new partition, specify the size of the partition in the space provided in MB and click Shrink to start the process.



3 Use the partition

After the process is complete, a new partition showing the amount of free, or unallocated, space appears next to the Windows C: drive partition. You can then point your Linux distro's installer to this free space. Remember to repeat the process and create another partition for the swap space as well.

Distro differences

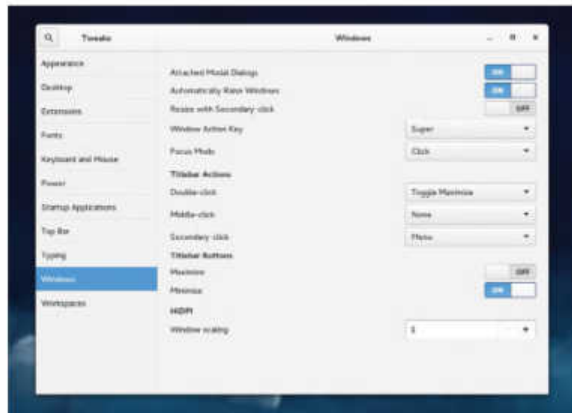
Taking baby steps with Linux.

» **O**ne of the biggest difference that foxes users coming from proprietary OSes is the lack of a consistent 'look and feel' to the Linux desktop. The default desktops on Ubuntu, Fedora and Mageia distros all look and behave differently from each other. We say default because unlike other proprietary OSes, a Linux distro enables you to swap out the desktop and replace it with an entirely different one to better suit your workflow.

The idea of the desktop as a separate entity from the operating system sounds foreign to users coming from Windows or Mac. But like all things Linux and open source, users are spoilt for choice when it comes to putting a face on top of their Linux distro.

Unity in diversity

The Ubuntu distribution uses its own home-brewed Unity desktop. The most prominent component on the desktop is the vertical Launcher which functions pretty much like a taskbar. It houses icons for the frequently used apps for quick access that you can modify as per your requirements. Also, some icons have specialised right-click context menus that give you quick access to frequently used features.



» **Users of Gnome-based distros should use the Gnome Tweak Tool to tweak the behaviour of their desktop.**

The first icon on the Launcher brings up the Dash, which is Ubuntu's take on the traditional menu-based navigation system. It features a search box at the bottom and anything you type here is used to look for matching apps, documents, music, videos, instant messages, contacts and other content. Furthermore, you can also use the Dash to install and uninstall apps and preview media files. Unity also includes the Heads Up Display (HUD), which is an innovative take on the application menus. Using HUD helps you avoid the trouble of looking for options embedded deep within nested menus. To access HUD press the Alt key from inside any app and use the Dash-like search box to perform a task.

The default Unity experience is the result of extensive usability research by Canonical. But you'll find some options to tweak the desktop from under the System Settings tool accessible via the gear & spanner icon in the Launcher. The settings are grouped into three broad categories. The Personal group houses settings for customising the look and feel of the desktop by changing the wallpaper and modifying the behaviour of the launcher. Pay attention to the Online Accounts settings which you can use to sign into several online services, such as Facebook and Google Docs, and integrate their contents with the desktop apps. For example, adding your Flickr account will integrate it with the Shotwell photo manager.

Gnome itself

Gnome is another popular desktop, and the Gnome 3 desktop contains more or less the same elements as Ubuntu's Unity but presents them in a different way. For starters the desktop is very bare. Click on the Activities button in the top-left corner to reveal the Overview which is very similar to Unity's Dash. In this view, you also get a Launcher-like Favourites bar for accessing frequently used apps.

In the centre you get a preview of all open windows. To the right is the Workspace Switcher, which always shows the current Workspace and an additional one. If you add windows

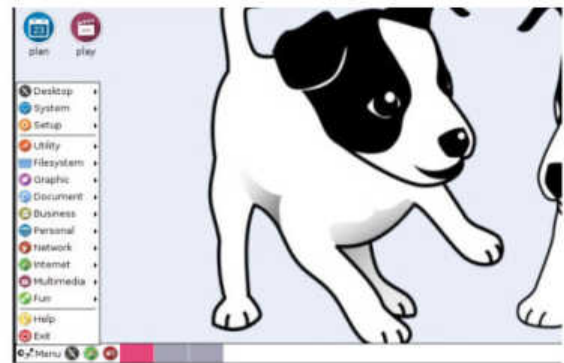
Resuscitate an old workhorse

One Linux speciality is infusing life into machines doubling up as paperweights because they can't keep up with the hardware demands of modern OSes. While there are many distros that are designed to power older hardware, our all-time favourite is Puppy Linux.

The distro uses one of the lightest window managers (JWM) and though it might not be pretty to look at, it'll turn that old lethargic work horse into a galloping stallion. But the main reason for Puppy's stellar performance on hardware with limited resources is its sheer number of lightweight custom apps. The distro has graphics apps, productivity apps,

apps to playback, edit and even create multimedia. Using its custom apps, you can block website ads, grab podcasts, do internet telephony, burn optical media, and a lot more.

The distro is available in different flavours. The Wary Puppy edition uses an older kernel and includes additional drivers to support peripherals like dial-up modems. There are flavours based on the recent Ubuntu releases too, such as Tahrpup based on Ubuntu 14.04 and Slacko Puppy based on Slackware Linux. These editions use a newer kernel than Wary but take advantage of Puppy's custom apps for limited resources.



» **Puppy Linux has very helpful forum boards and loads of documentation written specifically for new users.**

to the second Workspace, a third will automatically be added. At the top is a search box that will match any text to apps and documents on the local computer as well as online services. Gnome includes an Online Accounts app that enables you to sign into online services, such as Google Docs and Flickr. In fact, Fedora will ask you to sign into these online services when you boot into the distro for the first time.

The Gnome desktop also has several custom apps of its own that can fetch information and data from the added online accounts. For example, the *Gnome Contacts* apps can pull in contacts from various online sources, such as Gmail. Similarly, *Gnome Documents* will help you find documents from online repositories such as Google Docs.

New users should also keep an eye out for the desktop's peculiarities. For one, you won't find the Minimise buttons on any of the windows in Gnome. When you want to switch to another app, head to the Activities Overview and launch a new window or select an existing open one. Another esoteric aspect is the lack of any desktop icons as well as the ability to create any shortcuts or place any folders on the desktop.

However, Gnome's redeeming aspect is its tweakability: you can add new features literally with a single click. Gnome supports a plethora of extensions that you can enable without any installation. Just head to the Gnome Extensions website (<http://extensions.gnome.org>), find the plugin you wish to enable and toggle the button to activate it on your desktop. Some of the popular extensions are designed to help ease the transition for users moving to Gnome from proprietary operating systems, such as Windows.

Kick off with KDE

Unlike the other two desktops, the layout and behaviour of the KDE desktop and the placement of its Kickoff app launcher will certainly feel familiar to users from non-Linux operating systems. But KDE is so malleable that many KDE distros look unlike each other. In many ways, KDE is the quintessential Linux desktop with its flexibility and myriad number of options. There's literally no end to KDE's customisation options.

One of the most useful KDE features is Activities. Using this feature, you can create several context-aware activities, each with its own set of apps and desktop furniture. For example, you can create a Social activity that signs you into all your instant messaging accounts and displays updates and feeds from various social networks. Many KDE distros ship with just the default activity, called the Desktop Activity. However, you can fetch more activities from the internet and build on them to suit your workflow.

Furthermore, KDE ships with multiple interfaces or Views designed to make the best of the available desktop real-estate. There are different Views for regular screens and netbook though you can use any View on any type of computer. To switch Views, right-click on the desktop and from the context-menu select the Default Desktop Settings option. In the window that opens up, select the View tab and checkout the different views from the Layout pull-down list.

Many KDE distros place the Folder View widget on the desktop which displays the contents of a folder in a neat little box that you can place anywhere on your screen. Then there's the Folder View which lets you place files and folders anywhere on the desktop. The Search and launch View is designed for devices with a small screen or a touchscreen. Each View has additional configurable elements.

In addition to bundling the configuration options along with the individual elements, KDE also houses them all under the System Settings panel, alongside other system-wide configuration options to administer the underlying Linux distro. It might seem daunting, but you don't need to set up or review each and every option before using the desktop. Customising KDE is an on-going process and not a one-time

“Ubuntu, Fedora and Mageia are available in multiple editions with a different desktops.”

affair. The desktop is designed to grow and mutate as per your usage requirements.

31 flavours

In addition to these three chief desktop environments, there are a lot more that you can put atop your distro. There are fully fledged environments, such as Cinnamon, as well as lightweight ones, such as Xfce, LXDE and Mate. In fact, most mainstream distros, including Ubuntu, Fedora and Mageia are available in multiple editions with a different desktops.

For example, the Ubuntu distro has a number of officially supported spins. There's Kubuntu which dresses Ubuntu with KDE as well as a Gnome spin, a Xfce spin and another that uses the Mate desktop. The Fedora distro, which was once known as the premier Gnome distro, now also has a wonderful KDE flavour as well. Similarly, you can also use Mageia with the Gnome desktop as well. In fact, Mageia and Fedora, also have install-only DVD images that give the user the option to install multiple desktops.

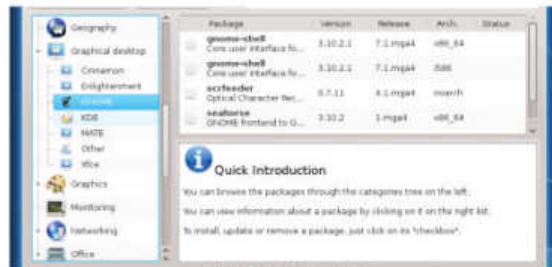
Switch desktop environments

You can also install multiple desktops on top of your distribution, such as the popular Cinnamon desktop. It's available in the official repositories of Fedora and Mageia and you can install it via their respective graphical package managers.

On Ubuntu, it's available via the **ppa:gwendal-lebihan-dev/cinnamon-stable** PPA. Add this PPA to your system [as explained in the main text] and then install the desktop environment from the *Software Center*.

Once you've installed multiple desktop

environments you can easily switch to another one. To do this you just log out of the current desktop environment, and use the login manager and enter your login credentials. Before logging into the desktop, explore the buttons on the login manager. One of the buttons will reveal a drop-down list of all the installed desktops. Select the desktop environment you want to use and the login manager will log you into that desktop. This way you can try them out and choose one you like the best. Choice!



» There are several desktop environments that you can install using your distro's package manager.

Install and manage apps

Fleshing out your chosen distribution.

» Unlike Windows, a typical Linux desktop distribution is more ready-to-use right out of the box. Instead of shipping with basic apps, such as a vanilla text editor or a barebones drawing tool, your average Linux distro will include a fully fledged office suite and a comprehensive graphics editor. This is in addition to the basic set of apps for common tasks, such as browsing the Internet, checking email, instant messaging with your friends across various networks, organising photos, listening to music and watching videos.

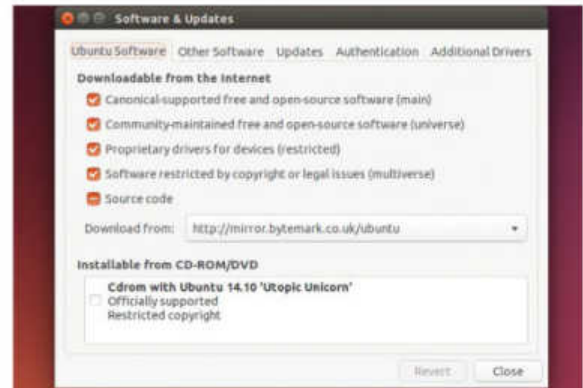
That said, since we all use our computers differently, you'll likely want a piece of software that isn't included by default. Your distro has specialised tools that'll help you install gazillions of quality open source software without clicking through complex setup wizards.

Linux distros use a collection of software tools, both graphical and command-line based, that are together referred to as a package management system. These tools help you install, remove, and upgrade software (also called packages) with ease. Individual pieces of software are grouped inside packages. In addition to the software itself, packages also include other information, such as a list of other packages or dependencies which are required for the app to function properly. Furthermore, the package management system relies on a database known as the repository to keep track of all the available packages.

Package Management 101

The Linux world is divided broadly into two different package formats – RPM and Deb. These are precompiled binary packages that are designed to simplify the installation process for desktop users. RPM was created by Red Hat Linux, and is used by distros such as Fedora, and Mageia while Deb is used on Debian-based systems, such as Ubuntu. Additionally, almost every major distro maintains its own set of graphical tools to enable desktop users to install, upgrade and remove app. You must also be familiar with the distro's repository structure and how and where it houses software.

Ubuntu uses the Advanced Packaging Tool or APT package



» In addition to enabling repositories you can also select a different mirror for downloading software.

management system. You can use the Software & Updates tool for manipulating Ubuntu's repositories (or repos). The tool lists repos in four different tabs. By default, the four official repos under the Ubuntu Software tab are enabled. The Main repo includes officially supported software, and the Restricted repo includes software that isn't available under a completely free license. The two interesting repos are Universe and Multiverse repos, which include software maintained by the community and software that isn't free, respectively.

Unlike Ubuntu the Fedora distro uses the RPM package management system. The distro houses repositories under the `/etc/yum.repos.d` directory and the main repository is named `fedora.repo`.

Mageia uses the `urpmi` package which is a wrapper for the RPM package management system. The distro has three official repos. The core repository contains open source packages, the non-free repository contains closed-source apps, and the tainted repository has packages that might infringe on patents and copyright laws in some countries. Each of these repos is further divided into four sub-repos. The release repo includes stable packages, the updates repo includes packages that have been updated since the release,

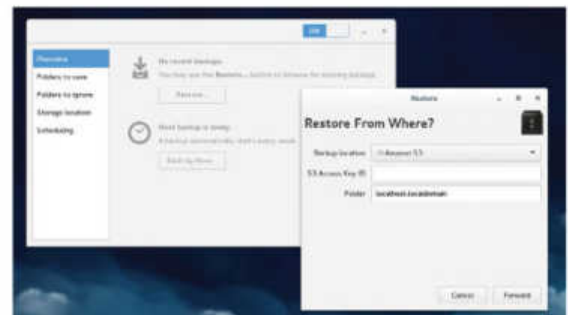
A backup primer

Your distribution will include a tool to help you backup your data and you should take some time out to get familiar with it. Ubuntu, for instance, ships with the *Déjà Dup* backup app which is designed for new users. You can also install it on top of Fedora and Mageia.

No matter what backup tool you use, you should take a moment to consider what you should backup and where. Backing up the entire **home** directory might be convenient but is usually just an overkill. Instead you should just include the directories under your home directory such as **Downloads** and **Documents**.

Also check with important app, such as email clients, who keep downloaded emails, attachments and address books under hidden directories (prefixed with a **✓**) beneath the home folder.

Also, keeping the backed up data on another partition of the same disk isn't going to be of much use, since the whole disk might fail and render the backup copy useless. One solution is to keep the backup on another separate disk or external drive. Or, if you have good Internet bandwidth, the backup app might also help you store the backups on a cloud storage service.



» *Déjà Dup* has a simple interface that shouldn't intimidate even first time users.

and the backports repo contains packages of new versions backported from the Cauldron repository, which will eventually become the stable repo for the next release. There's also the testing repo which will contain software primary for QA purposes.

To configure repos, launch the *Mageia Control Center* and head to Software management > Configure media sources for install and update. To install the official online repositories, click on Add and then either on Update sources only or Full set of sources. The first choice is the minimum to keep the distro updated, while the second allows you to install new software. These options will populate the window with a list of repos. Then toggle the Enabled checkbox next to the repo you want to fetch software from.

FedEx packages

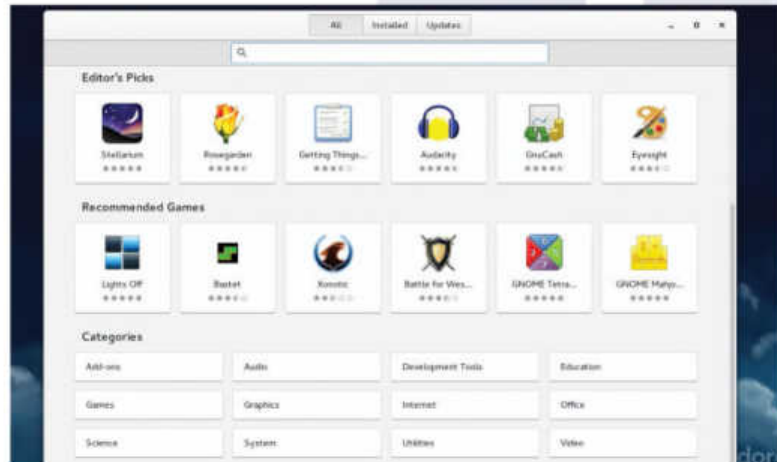
All major desktop distros include a graphical tool for managing packages. Ubuntu's *Software Center* is one of the best tools for the job. You can find software by clicking on the category reflecting the type of software that you're looking for. When you select a category, you will be shown a list of apps. There's also a search box in the upper-right corner of the window which will look for software matching any entered keywords. Once you've found the software you want, click the Install button to its right. This will fetch the software as well as any required dependencies and automatically install it. All newly installed software is added to the Launcher and you can also find it from under the Dash.

Fedora uses the *PackageKit* graphical tool that's listed as Software in the Gnome's Activities menu. It too lists software categories as well as a keyword-matching search box at the top to help you find software. Once you've found the software that you're looking for, click on the Install button and the app will add it to your installation.

The graphical package management tool for Mageia is named *Drakrpm*. The tool isn't as pretty as the software centres in Ubuntu and Fedora, but is very functional and intuitive enough to get the job done. You can filter its list of available apps to show only packages with GUI, security updates, bug fix updates, and more. Applications groups are listed in the sidebar and there's also a search box to hunt for packages based on keywords. When you find a package you wish to install, simply toggle its corresponding checkbox and click on Apply.

The Repo men

The larger open source community offers a lot more packages than the ones listed in your distro's official repos



and almost every distro has a mechanism to add and install software from these third-party repos.

External repos in Ubuntu are known as a Personal Package Archive or PPA. You can add a PPA repo to your distro using the Software & Updates tool. But first you need the address of the PPA. This is listed on the PPA's Launchpad site and will be something like `ppa:example-ppa/example`. Now fire up the tool and switch to the Other Software tab. Then click on the Add button and paste the address of the PPA in the window that opens. Ubuntu will then ask you to refresh the repos to enable the PPA.

» **Most desktop distros have an easy to use graphical package manager.**

“Unlike Windows, a typical Linux desktop distro is ready-to-use right out of the box.”

Similarly, Mageia has a number of third-party repos as well and you'll need their URL to add them to your distro. Once you have the URL, fire up the *Mageia Control Center* and head to Software management > Configure media sources for install and update and click on the Add a medium option. Enter the address of the repo in the window that pops up along with its type such as HTTP or FTP.

Fedora also has a number of third-party software repos but the most popular is RPMFusion. The repo is further subdivided into two independent repos that house free and non-free software. You can install both of these repos from within the browser itself by following the instructions on the repos website (www.rpmsfusion.org/configuration).



Grabbing Google

All popular Google software, such as *Chrome*, *Earth*, the audio and video plugin for *Hangouts* and others can be installed on Linux. But you won't find them in the official repos of major distros because of their licensing. However, you now have all the know-how to install them with ease if we point you in the right direction.

The downloads page of each supported Google app contains links to both 32-bit and 64-bit versions of the software in both RPM and Deb formats. Download the package for your

particular distro and double-click on the file to install it with the distro's package manager. All official packages for Google apps will also install the appropriate external Google repository in your distribution to keep the software updated.

Another popular proprietary software that you may want to install is *Skype*. Ubuntu users can simply enable Partner Repositories by visiting Software & Updates > Other Software. This will add the official *Skype* repos and you can then install the software from the *Software Center* as

for any other software package.

Mageia, on the other hand, includes the *Skype* package in its non-free repo. If you've enabled this repo, then simply search for the **get-skype** package which will download *Skype* from its website. You can also head to the Linux download page on *Skype*'s website and choose your distro and architecture from the pull-down list which will download either a Deb file or an RPM file. Double-click on the file to install it with the distro's package manager.

Play multimedia

Turn your distro into the ultimate media centre.

» Most Linux distros designed for desktop users are capable of handling all types of content you throw at them. But some content, especially most audio and video, is distributed in closed formats that are encumbered by patents. The distros can't play these files straight out of the box, however, most have clearly outlined procedures to allow users to install components to play these popular non-free media formats.

Ubuntu gives you the option to install the components that can play files in restricted formats, such as MP3s, during the installation itself. If you've already installed the distro, you should then use the package manager to install the **ubuntu-restricted-extras** package, which includes popular proprietary codecs and plugins.

On Fedora these codecs are bundled in the third-party RPM Fusion repository. You'll have to first enable the repo as mentioned earlier and then fire up a terminal and enter the following commands to fetch the codecs:

```
su -
yum install gstreamer{1,}-{plugin-crystalhd,ffmpeg,plugins-
{good,ugly,bad}{,-free,-nonfree,-freeworld,-extras}}{-extras}}
ffmpeg libmpg123 lame-libs
```

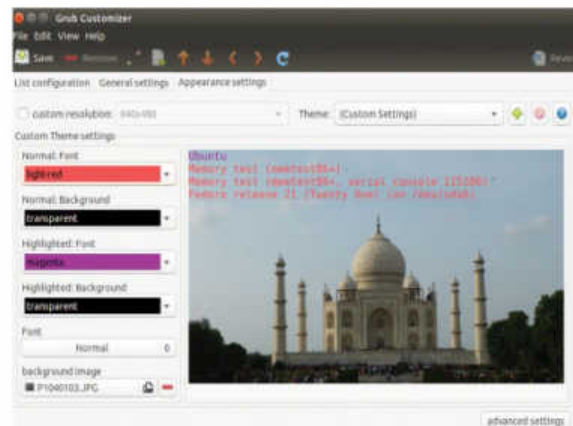
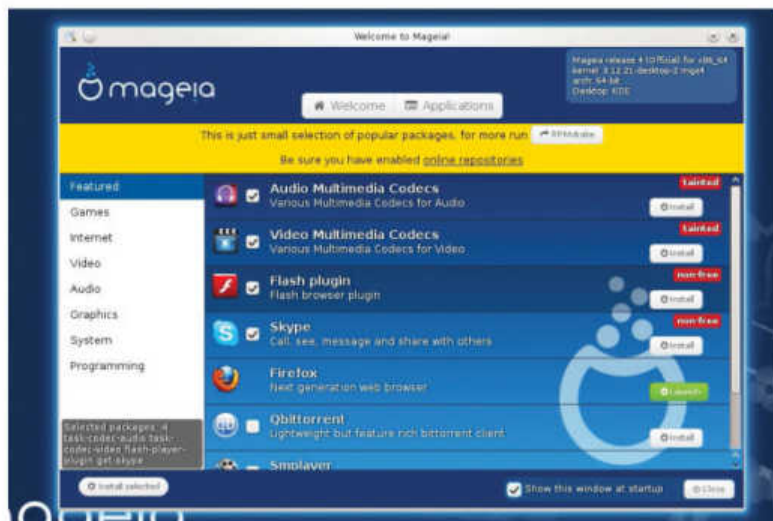
If you're using Mageia, you'll find the multimedia codecs under the Tainted repository, so make sure you enable it following the procedure mentioned earlier. Then launch the *Welcome* app from under the Tools menu and switch to the Applications tabs. From here you can install several useful and popular packages including multimedia codecs.

Where's the bling?

Despite the rise of the open source WebM format, many websites still require Adobe's Flash plugin to properly stream multimedia content. Getting the Flash plugin on Linux is tricky since Adobe is no longer developing Flash for *Firefox* on Linux. The only way to use the latest Adobe Flash plugin on Linux is to use Google's *Chrome* browser which includes the Pepper-based Flash plug-in.

That said, if you don't want to switch to *Chrome* you can

» Mageia's *Welcome* app is a wonderful utility to setup the distro for all kinds of users.



» You can tweak Linux's Grub bootloader with the **Grub-Customizer** tool ([ppa:danielrichter2007/grub-customizer](https://ppa.danielrichter2007/grub-customizer)).

still install the out-of-date Flash plugin and continue using the *Firefox* browser. Or, you can extract the newer Pepper-based Flash plugin from the *Chrome* browser and use it on *Chrome*'s open source cousin, the *Chromium* browser.

Ubuntu users can install Flash for *Firefox* with the

```
sudo apt-get install flashplugin-installer
```

command. If you're using *Chromium*, you can use the latest Pepper Flash plugin by installing the **pepperflashplugin-nonfree** package.

Fedora users can download the *Firefox* plugin from Adobe's website by adding its repo. If you are running a 64-bit installation, this command:

```
yum -y install http://linuxdownload.adobe.com/adobe-
release/adobe-release-x86_64-1.0-1.noarch.rpm
```

will download and install the correct repo. You can then import the key for the repo with:

```
rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-adobe-linux
```

```
yum -y install flash-plugin
```

Mageia users can simply enable the **nonfree** repository which houses the Flash plugin and then install it from the Applications tab in the *Welcome* app.

Best multimedia apps

Most distros ship with audio and video players. The popular ones are *Rhythmbox* which is the default on Gnome-based distros and is well integrated in Ubuntu, and KDE's default *Amarok*. In addition to local tracks, both players can also stream Internet radio and podcasts. If you want more attractive looking players, fire up the package manager and look for *Banshee* and *Clementine*.

Similarly, the default video player on most Gnome-based distros is *Totem* (now simply called *Videos*). If you want something with more feature you can grab *MPlayer*. This is essentially a command-line media player but it has a number of frontends. Gnome users can use *Gnome-Mplayer* and KDE users can use *KMPlayer*. There's also the popular cross-platform *VLC* that can handle pretty much any file format.

Play games

Full steam ahead.

Gaming has long been considered Linux's Achilles's heel. Over the years we've had several quality open source games, but they've lacked the mass-appeal of popular gaming titles available on proprietary desktops. All that changed in early 2013 with Valve's announcement of *Steam for Linux* client for its hugely popular game distribution service.

You can install the proprietary client in your distro with ease. Ubuntu users should enable the Partner repo and then install the client from the *Software Center*. Fedora users should install the RPM Fusion repos and then install *Steam* using the package manager. Similarly, Mageia users should enable the non-free repo and then use the Mageia *Welcome* app to install the *Steam* client.

But before you fire up the *Steam* client make sure you've got the proper drivers for your graphics hardware. This tends to be the major sticking point for users, as card manufacturers restrict distribution of their closed-source driver elements. Start by getting details about the make and model of your graphics card using the

```
lspci | grep VGA
```

command. You can get more details, such as its clock speed and capabilities, with:

```
sudo lshw -C video
```

Your distro will ensure you are using the most suitable open source driver as soon as you boot into the distro. The Oibaf PPA ([ppa:oibaf/graphics-drivers](https://ppa.launchpad.net/oibaf/graphics-drivers/ubuntu)) is popular with Ubuntu users for getting the latest bleeding edge open source drivers. However, users of Nvidia and ATI/AMD hardware should use proprietary drivers from the respective vendor for the best gaming performance.

Ubuntu users should use the X-Swat PPA ([ppa:ubuntu-x-swat/x-updates](https://ppa.launchpad.net/xswat/x-updates/ubuntu)) for the latest stable Nvidia drivers. Once enabled fetch the drivers with:

```
sudo apt-get install nvidia-current
```

Fedora users will find the latest GeForce drivers in the RPM Fusion repo. After adding the repo, install the driver with



```
yum install kmod-nvidia xorg-x11-drv-nvidia-libs kernel-devel acpid
```

Mageia users should first enable the non-free repo and then launch the Mageia Control Center and head to Hardware > Set up the graphical server. In the window that open click on the toggle next to the Graphic Card label which will display a list of graphics card. Browse the list and select yours. If Mageia has a proprietary driver for the card, it'll install it.

Mageia's list also includes AMD cards, but if you are using Ubuntu or Fedora, the best source for the proprietary driver is AMD's website (<http://support.amd.com/en-us/download>). This page has several dropdown menus that you can use to pinpoint the exact driver for your graphics card. Then download the suggested driver and extract it to reveal a **.run** script. Before you install the driver make sure you install its dependencies with:

```
sudo apt-get install dh-make dh-modaliases execstack libcc-1386 lib32gcc1
```

Once that's done, you can execute the script with:

```
sh ./amd-driver-installer-13.35.1005-x86.x86_64.run
```

This will launch the graphical AMD Catalyst proprietary driver installer and will also install the *Catalyst Control Center* GPU management software. When the installer has finished, head back to the terminal and enter

```
/usr/bin/aticonfig --initial
```

to configure the driver.

While distros do come with open source games in their repos, Steam offers access to AAA gaming titles like *Dying Light*.



User admin basics

When Linux is installed, it's automatically configured for use by a single user, but you can easily add separate user accounts. The superuser, root, has complete access to the OS and its configuration; it's intended for administrative use only.

Unprivileged users can use the **su** and **sudo** programs for controlled privilege escalation. Users are grouped together into a group, and inherit the group's access privileges. Every Linux user is a member of at least one group. Now, you can control which files and folders are accessible by a user or a group. By default, a user's files are only accessible by that user, and system files are only accessible by the root user. In Linux files

and folders can be set up so that only specific users can view, modify, or run them. This allows you, for instance, to share a file with other users so they can read the file but not make any changes.

Access to files is controlled via a system of ownership permissions. You can use the **ls -l** to list the permissions. When used without specifying a filename it will list the permissions of all files within the directory. The different characters in the output represent a file's read (r), write (w), and execute (x) permissions for the owner, group, and all other users. You can alter the permissions of files and folders with the **chmod** command or graphically from the file manager.



The file manager of almost every distro allows you to tweak the group associated with a file or folder.

GIZMODO

UK

Not your average technology website



EXPLORE NEW WORLDS OF TECHNOLOGY GADGETS, SCIENCE, DESIGN AND MORE

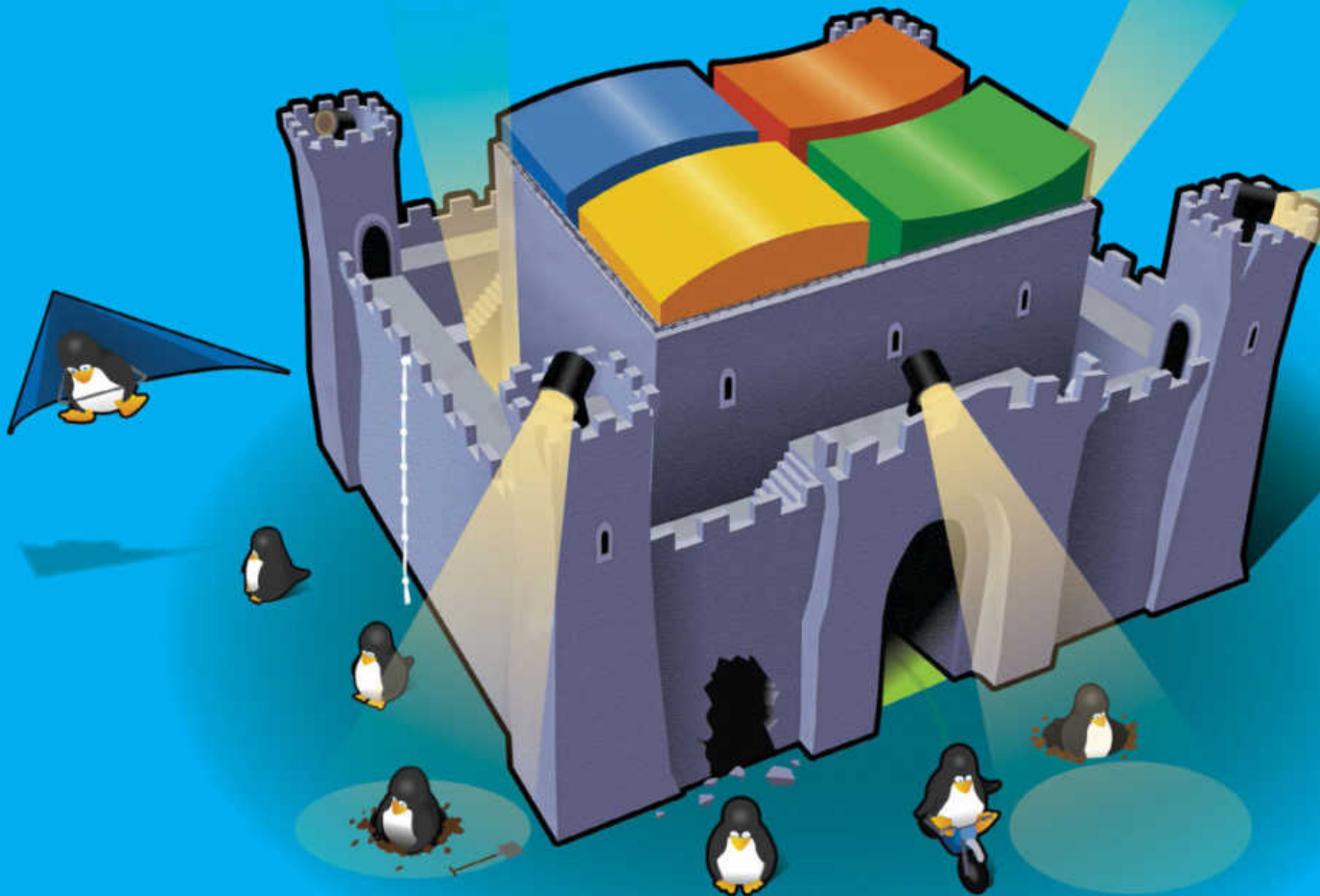
- Fascinating reports from the bleeding edge of tech
- Innovations, culture and geek culture explored
- Join the UK's leading online tech community

www.gizmodo.co.uk

twitter.com/GizmodoUK facebook.com/GizmodoUK

Escape Windows

The time has come – follow us as we help you evade Windows and make a bid for Linux and freedom.



Get started | Escape Windows

If, having bought a new PC that has Windows pre-installed, you've come to the conclusion that Microsoft has finally lost the plot, now is the perfect time to switch to a better OS. Macs are shiny and expensive, so why buy one when you can keep your existing PC and move to Linux? Gone are the days when Linux meant immersing yourself in command-line chicanery – we'll show you how you can install a user-friendly flavour of Linux alongside your existing Windows installation without losing any data, so you can dip your toes into the free and open waters of Linux.

To make the transition as smooth as possible, we've chosen Ubuntu 14.04 LTS (on

the cover disc) as our Linux distro of choice. Ubuntu is one of the most user-friendly Linux distros out there, and 14.04 LTS has just been released. LTS stands for Long Term Support, which means it gets five years of updates and support.

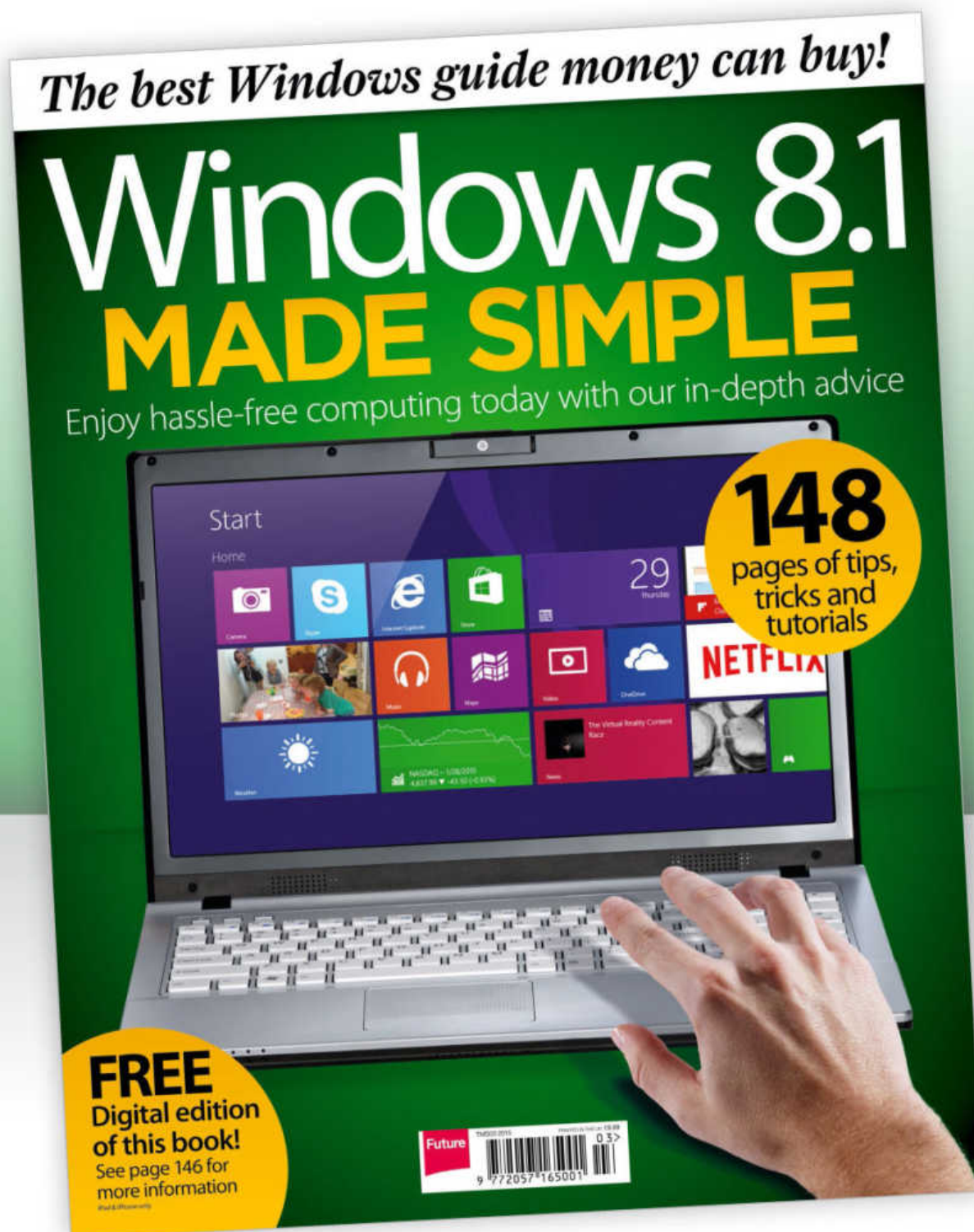
“Microsoft has finally lost the plot – now is the perfect time to switch to a better OS”

Here's what you're going to do. Firstly, we would be remiss if we didn't instruct you about how to back up your current PC. You have two basic choices here: the back-up tools which are built into Windows itself, or SystemRescueCD, which you'll find as

part of the cover disc. Next, we'll explain how modern PCs ship with a new system called the UEFI, which may cause minor problems. But don't worry, because we'll show you how to overcome any issues if they arise. Finally, we'll step you through the installation process itself, which is a piece of cake. And we won't leave you hanging – once you've booted into Ubuntu for the first time, we'll take you on a tour of the new desktop, reveal how to access existing documents and other files, and show you how to get to grips with your brand new operating system. There's never been a better time to make the life-changing switch to Linux, so grab our hand and take the plunge.



YOUR COMPLETE GUIDE TO USING WINDOWS!



**OUT
NOW!**
WITH
FREE
DIGITAL
EDITION



DELIVERED DIRECT TO YOUR DOOR

Order online at www.myfavouritemagazines.co.uk
or find us in your nearest supermarket, newsagent or bookstore!

Back it all up!

Stick with us – before migrating to Linux, make a fail-safe backup.

The simplest way to back up your system is to use the built-in Windows Backup tool in Windows 7 or later – select 'System Image Backup' and follow the prompts. It's not very exciting and it's a little scary, but it is essential. For a pure Linux back-up solution, use SystemRescueCD, which is on the cover disc. Here's how...

Reboot your PC with the disc in your drive, access the boot menu if necessary (usually [F11]) and choose your DVD drive, then 'SystemRescueCD'. Problems booting could be caused by your PC's UEFI (see over the page to fix them). Otherwise, pick the default boot options entry. Type **uk** when prompted for the keyboard layout, and type **startx** at the final command prompt to access the graphical environment.

Once this loads, click the 'Menu' button in the bottom-left and select 'System > Show Filesystems' to identify each partition. The Disk entries identify each physical drive (sda, sdb and so on) and the Device identifies the partitions. In most cases, sda will be your Windows drive, so you'll need to back up sda1 (a small boot partition) and sda2 (the actual Windows partition). You'll also need to note the partition you intend to back up to and make sure it has enough free space.

Now switch to the terminal window. To mount the back-up partition, type the following command, changing **ntfs** to **FAT** if your back-up drive is FAT32, and replacing **sdcd2** with the drive's actual partition name:

```
Mount -t ntfs /dev/sdc2 /mnt/backup
```

Now re-open the menu and select 'System > PartImage'. This tool lets you back up partitions, so start with **/dev/sda1**, then press [Tab]. Next, type **/mnt/backup/sda1-backup.gz** into the 'Image file to create/use' box. Leave 'Save partition into a new image file' selected and press [F5]. Verify 'Gzip' is selected under 'Compression level', then click [F5] again. Enter a description if you wish, then press [Tab] to select 'OK'

and hit [Enter]. Make a note about experimental NTFS support, then click 'OK' twice, noting the drive statistics.

If this successfully completes, repeat to back up your Windows partition, naming it **sda2-backup.gz** instead.

To back up your computer's Master Boot Record, open a terminal window and type the following commands

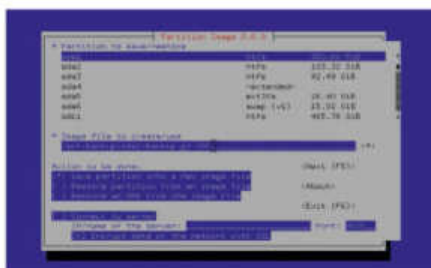
```
cd /mnt/backup
mkdir partition-backup
cd partition-backup
dd if=/dev/sda of=backup-sda.mbr count=1 bs=512
to back up MBR. Now back up your partition table with:
sfdisk -d /dev/sda > backup-sda.sf
```

If you get a warning about *sfdisk* not supporting GPT partitions, type **sgdisk --backup=backup-sda.sg /dev/sda** instead. If disaster strikes, either boot from your Windows rescue disc and follow the prompts, or boot back into SystemRescueCD. If you got as far as Ubuntu resizing your Windows partitions, you'll first need to launch the *Gparted* partitioning tool from its Taskbar shortcut. It should display your main hard drive (**/dev/sda**), so identify the extended partition. Right-click each volume inside it and choose 'Delete', then right-click the extended partition and delete that too. Next, right-click your Windows partition and choose 'Resize/Move'. Allocate it all available space, click 'Resize/Move' then 'Apply'. To complete the partition restore process, follow the step-by-step guide (below). The final step is restoring the MBR. Open another terminal window and type:

```
cd /mnt/backup/partition-backup
dd if=backup-sda.mbr of=/dev/sda
```

If you backed up your partition using *sfdisk*, type **sfdisk /dev/sda < backup-sda.sf** into the command prompt; if you used *sgdisk*, type **sgdisk --load-backup=backup-sda.sg /dev/sda** instead to complete the restore. »

Restore your original setup



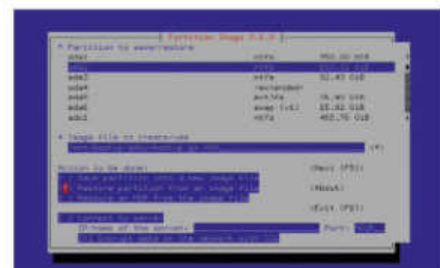
1 Select backup

Boot into SystemRescueCD as before. Follow the steps in the main text (above) up to identifying your partitions. Next, mount the back-up partition in the **/mnt/backup** directory. Once done, open *PartImage*. Leave 'sda1' selected, type the back-up path and name from before into the 'Image file' box.



2 Restore boot partition

Press [Tab] twice to highlight the asterisk by 'Save partition', then press the down arrow then [Space] to select 'Restore partition from an image file'. Press [F5]. Note the description and press [Enter]. Press [F5] to restore the partition, working through the warnings until the files are copied back.



3 Restore main partition

PartImage will close once the partition has been restored, so launch it again but this time select your main Windows partition (typically 'sda2'). Then it's a case of repeating the previous steps, only this time using the first back-up file. Note: the restoration relies on your Windows partition being the same size.



Handling UEFI & Ubuntu

Get your system configured correctly, even Windows 8 systems.

Modern PCs, particularly those with Windows 8 pre-installed, have replaced the traditional BIOS with a new system called UEFI. Both perform the same job, detecting and initialising your hardware before handing it over to your OS, but UEFI represents a quantum leap forward from the somewhat primitive capabilities of the traditional BIOS. The UEFI offers a much wider range of features than that found in the BIOS, all wrapped up in a much more user-friendly, graphically pleasing interface.

There are, however, potential issues to circumvent. First, because UEFI is a newer standard, it's not compatible with legacy hardware. This problem is circumvented by a special Compatibility Support Module (CSM), which enables you to emulate the BIOS to run older hardware and OSes.

More controversially, UEFI also ship with a Secure Boot mode, which, while making PCs less susceptible to malware, were used to lock them to Windows 8. Again, this feature can – in most cases – be disabled, but the good news is that Ubuntu 14.04 LTS fully supports newer hardware and Secure Boot, so you won't need to disable either to install it alongside



▶ **Make sure your FastBoot settings are disabled before attempting to install Ubuntu.**

Logo Requirement. Either way, disable it if you have problems booting SystemRescueCD. If the section isn't present, look under Security instead.

The CSM module may be part of the Windows 8 Logo Requirement screen, or it'll be hidden away on its own (we found it buried away at the bottom of the Boot menu on our ASRock motherboard, for instance). Click this to reveal a selection of different options covering different hardware – look for references to PXE (pre-boot environment), GOP (graphics) and Storage (or SATA), as well as USB and PS2, if applicable. This enables you to selectively switch CSM on and off for specific devices as required.

Some motherboards – including our ASRock – enable you to run both modes as required. (In the case of ASRock, we set each one to 'Do not launch' to make this setting the norm.) If there's no apparent options for editing the Secure Boot or CSM settings, contact your PC manufacturer to verify whether or not these can be disabled. In most cases, PCs built using off-the-shelf motherboards should have the required options intact, but some manufacturers have been known to provide hobbled boards.

Upgrading a PC's graphics card often requires enabling CSM support, so even if the controls aren't already supported, the manufacturer may have relented under

“Because UEFI is a newer standard, it's not compatible with legacy hardware”

Windows. One UEFI feature you'll need to disable, however, is FastBoot (also called QuickBoot or FastStartup).

Tweak the UEFI

Accessing the UEFI is the same as entering the BIOS setup. Switch on your PC, then tap the key prompted to enter setup when it appears on-screen (typically [F2] or [Del]). Verify that FastBoot has been disabled by looking for a section called Windows 8 Configuration, or – if it's not present – Boot. Once located, make sure it's switched off or disabled.

You may need to tweak the CSM and Secure Boot settings if you intend backing up your system using SystemRescueCD on the cover disc. Again, look in the Windows 8 Configuration, if it exists, for Secure Boot, which may be listed as Windows 8

Choose the right boot option

Many modern PCs now give you the option of booting either in UEFI or legacy BIOS mode, with the latter mode enabling you to install older operating systems. Ubuntu works fine with both UEFI and Legacy modes, but other environments – including SystemRescueCD – require booting from a legacy option.

Some PCs even support mixed-mode booting, presenting both legacy and UEFI choices direct from the boot menu, so you don't

need to wade into the UEFI shell to switch between them. If you've wondered why some boot options are displayed twice in the boot menu, this is why: look more closely and you'll see one option is prefixed by UEFI. This gives you the option of choosing which mode to boot that drive in, which is a good way of testing whether or not a certain OS or bootable disc works with UEFI or not. If you're only offered a single set of boot options, your UEFI shell will

only allow you to boot in either UEFI mode or legacy. The latter requires that CSM is enabled, but once done you should be able to change boot settings – including device priority – from within the Boot section of the UEFI shell. If your PC does support mixed-mode booting, you may need to set a different default boot device, in which case take care to select the right mode when choosing which device (typically optical drive or boot hard drive) to make the default.

pressure from end users and provided a UEFI update on its website to enable the required options – Advent's DT2410 PC is an example where this happened..

With your Windows system backed up and the UEFI configured correctly, it's time to press on and install Ubuntu alongside your existing setup. One of the reasons we've chosen Ubuntu is that its user-friendliness is apparent from the moment you install it. The set-up process is painless and relatively straightforward to follow: pop your cover disc into the drive, start your PC and, if necessary, select your optical drive from the boot menu. When the cover disc menu appears, choose the Ubuntu option to begin.

Wait for a few minutes, then after a short pause, when the screen goes grey and then red, the Welcome screen will appear. You'll be given the option of either trying Ubuntu or installing it – select the 'Install' option. Next, tick both boxes: 'Download updates while installing' and 'Install this third-party software'. Click 'Continue', then wait while the updates are downloaded in the background.

Install Ubuntu 14.04 LTS

The Installation Type screen should then appear. In most cases, it should seamlessly detect your Windows installation – leave 'Install Ubuntu alongside...' and click 'Continue'.

If Windows isn't offered as an option, this means Ubuntu hasn't detected it. If you're running a newer PC with an UEFI installed in place of the traditional BIOS, you'll need to make sure you boot from the optical disc using the same option that Windows was installed with. Reboot your PC, then choose the alternative optical disc option from the boot menu to see whether Ubuntu now detects your Windows installation. If the option doesn't appear whether you choose UEFI or the legacy option, it could be down to your partition table containing both MBR and GPT records (see below for the step-by-step guide for a way to resolve this issue).

The next screen sees Ubuntu automatically find some free space for itself – either by carving up your current Windows partition, or by locating more space on another attached drive and repartitioning that instead. If you don't want it installing on a USB drive, power down your PC, disconnect the offending drive, then start again. Ubuntu needs a



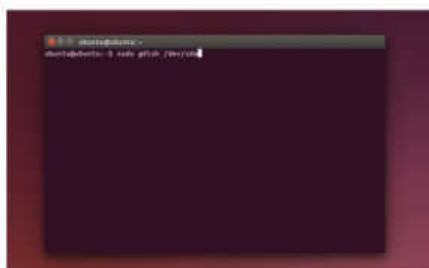
► Check your UEFI shell's boot settings to see whether Legacy, UEFI or mixed mode is selected.

minimum of 6.4GB of free space, but will take up to 40-odd GB if it can. You can alter this figure by clicking and dragging the grey space between both suggested partitions. It's smart enough to not allow you to swallow up your entire Windows partition, but if you plan to continue using it going forward, try to leave at least 10GB of free space if you can, and preferably more. In most cases, the default choice should be fine anyway, so click 'Install Now'. Note the comment about partition resizing, and click 'Continue'.

Next you'll be prompted to choose your region – the installer should automatically detect this – then click 'Continue' to select your keyboard layout. In most cases, the default UK keyboard layout is the one to pick, so click 'Continue' again. You'll now be prompted to enter your name. As you enter it, Ubuntu will populate both the computer's name and username fields based on your choice – you're free to edit both should you choose. The final choice available to you is to pick a password. Select one that you'll easily remember, because you'll need this not just to log into Ubuntu but also to perform administrator-level tasks. Click 'Continue' again, and you can sit back and wait for Ubuntu to install itself on your PC – it really is that simple. It's worth hanging around because the installer will reveal some useful pointers for your first steps into using Ubuntu. But turn the page and we'll take you on your own personal tour of your shiny, new desktop and operating system.



Fix Windows detection issues



1 Launch Ubuntu Live

If your Windows install isn't detected, click 'Back' until you return to the Welcome screen. Click 'Try Ubuntu' to load the live CD. When it's loaded, press [Ctrl]+[Alt]+[T] to open a terminal window. Type the following command and hit [Enter]:

```
sudo gdisk dev/sda
```



2 Verify and fix

If *gdisk* says it's found both an MBR and a GPT, then the presence of GPT is the problem – removing it will enable Ubuntu to see your Windows partition. First, select the GPT entry by typing **2** and hitting [Enter]. When prompted, type **x** and hit [Enter] to access Expert command mode.



3 Apply and reboot

Type **z** and hit [Enter]. When prompted to wipe the GPT on **/dev/sda**, type **y** and hit [Enter]. When asked if you'd like to blank the MBR, make sure you type **n**, hit [Enter] and close the terminal, then reboot and try to install Ubuntu again. This time you'll see the option to install it alongside Windows.



Get started with Ubuntu

Take your first steps into the Unity desktop.

You've installed Ubuntu, removed the cover disc and rebooted your PC. The first change you'll see is the presence of a new boot menu on startup, called **GNU Grub** (see 'All about Grub', p19). This enables you to choose which OS to boot into, plus gives you access to a memory testing utility. The default choice is Ubuntu, but before booting into it for the first time, we recommend selecting the **Windows** entry. This lets Windows boot for the first time since the Ubuntu installer resized your partitions – you'll note it immediately runs the disk checking tool to verify and update its own partition records. Let it do this, then once complete, it'll restart your PC. Select 'Windows' again from the Grub boot menu to check all is well. Assuming it is, restart your PC again.

This time, leave 'Ubuntu' selected and hit [Enter] to boot into your new OS for the first time. After the loading screen flashes by, you'll be prompted to enter the password you set up during installation. Type this in, hit [Enter] and you'll arrive at Ubuntu's desktop, which is called **Unity**.

The Unity desktop shares a number of similarities with Windows, such as the presence of shortcut icons on the left, but you'll also notice some key differences, which will feel more familiar to those who have used a Mac before. First, the

Menu Bar at the top of the screen works in a similar way to the Windows Taskbar Notification area, with a number of icons providing access to key tools. From the off you'll see – from left to right – network, keyboard, volume, date/time and settings. Most of these are self-explanatory – simply click one to reveal its Options menu.

The 'Settings' button is the most interesting of these – not only does it provide access to the power options, but it also enables the ability to quickly switch users, log out or lock the computer, and it also provides a handy shortcut to Ubuntu's System Settings, more on which in a moment.

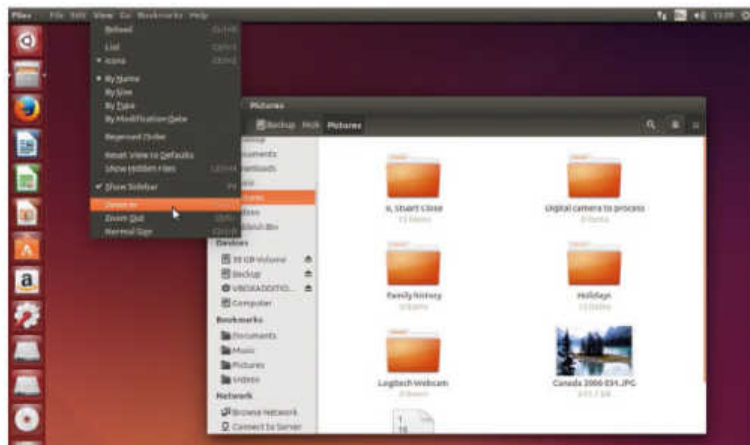
When you open an app, its title will also appear in the Menu Bar – roll your mouse over the Menu Bar and its menu items will appear, and clicking one will open its menu.

The App Launcher

The shortcut icons displayed on the left-hand side of the screen are part of Unity's App Launcher. Rather than cluttering up your desktop with lots of icons, these are all housed in a single column – simply roll your mouse to the bottom or top of the launcher to reveal more shortcuts. The App Launcher houses shortcuts to applications, workspaces, removable drives and the rubbish bin. The two shortcuts at the top, however, are of particular interest: the top one enables you to quickly search your computer and the internet from one handy window, while the Files shortcut gives you access to a file manager, making it easy to browse, access and manage your files – again, more on that later.

To find out more about a shortcut, roll your mouse over it and its title will pop up. Click one to open it. As time moves on, you'll want to know how to add and remove shortcuts from the App Launcher – to remove a shortcut, simply right-click it and choose 'Unlock from Launcher'. The simplest way to add a shortcut is to use the search tool to locate the app in question, then simply click and drag its shortcut into place on the Launcher. You can also reorder items in the Launcher by simply clicking and dragging them to their new spot.

The desktop itself works in a similar way to that of Windows, and can be used to house files, folders and shortcuts should you wish to clutter it up – again, it uses the familiar right-click menu to provide you with options for



▶ The main distinction for former Windows users trying out the Unity desktop environment for the first time will be the position of the launch bar.

Get more apps

Ubuntu 14.04 LTS ships with a few apps pre-installed – notably *LibreOffice* and the *Firefox* web browser. For a complete list of installed apps, open the Ubuntu Software Center and select the 'Installed' button. Expand a category to see what apps are there, such as *Rhythmbox*. Use the Search box to locate them, then either open them directly from here or drag a shortcut on to the App Launcher. There are many more apps you can add to your Ubuntu installation,

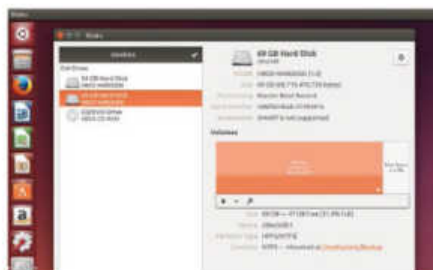
and the vast majority are free. There's a number of ways to do this: as you've seen with *Ubuntu Tweak* (see 'Accessing your documents', above-right), you can download apps from the web or install them from our cover disc. If they're packaged as Deb files, you can double-click them and have the Ubuntu Software Center manage installation.

You can also search for more apps from within the Software Center, too – just use the

search box. The Ubuntu Software Center isn't a definitive source for apps, however – Linux apps are housed in special collections called repositories, and as you'll see over the page, you can install apps via the terminal, too.

One thing you'll want to do is access apps for specific file types, such as images. If you can't find an app, try double-clicking a file you want to open – if you get a 'Could not display' error, click 'Yes' to search the Ubuntu repos for apps.

Accessing your documents



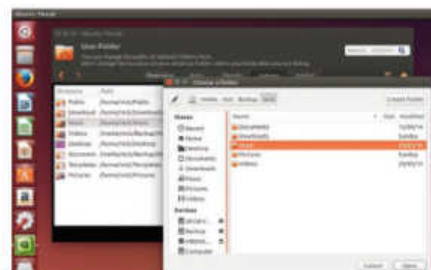
1 Identify mount point

Click 'Search' in the App Launcher, type **disks** and click the 'Disks' shortcut under 'Applications'. Assuming you've moved your data to a new drive, select this drive in the left-hand pane. Now highlight the partition that your data is stored on under 'Volumes' and make a note of where it's mounted.



2 Install Ubuntu Tweak

Open the *Firefox* web browser and browse to <http://ubuntu-tweak.com>, ignore the Launchpad prompt, then click the latest Deb Package link (0.8.7 at time of writing). Click 'OK' to launch it with Ubuntu Software Center, then review the package description before clicking the 'Install' button.



3 Move user folders

Click the 'Search' button, type **Ubuntu** and select 'Ubuntu Tweak' under 'Applications'. Select 'Admins > User Folder'. Each folder will be listed – select one and click 'Change'. Browse your data partition for the corresponding folder, select it and click 'Open'. Repeat for all the other user folders.

creating folders, managing existing icons and changing the desktop background, should you wish.

Ubuntu can detect both FAT32 and NTFS partitions, so accessing your old data shouldn't be too much of an issue. You should see these drives appear in the App Launcher – clicking one will open its contents for viewing. If you're looking for your personal data, unless you've specifically moved the folders in Windows, you should find all your documents in the Users folder under your username.

Access your data

You can also access the graphical *Nautilus* file manager by clicking the 'Files' shortcut on the Launcher. It provides a familiar two-paneled window, with a series of shortcuts on the left and the contents of the currently selected folder or drive on the right. By default, items are displayed in a grid, but you can switch to list view using the buttons in the top-right. You'll also see a 'Search' button, which enables you to search the currently selected folder (and any sub-folders).

If you want to copy data from your Windows partition to your user folders in Ubuntu, browse to one of your user folders – **Documents**, for example – then click and drag all of its contents to the 'Documents' shortcut in the left-hand pane. It pays to make sure you have enough free drive space. If you plan to continue using Windows alongside Ubuntu, a better solution is to set up a spare drive or partition as a dedicated data drive, allowing both Ubuntu and Windows to access the same copy. This needs to be a FAT32 or NTFS partition so Windows can access it as well as Ubuntu.

Assuming such a partition is in place, set it up by first booting back into Windows. Now create dedicated folders for **Documents, Pictures, Music and Videos** on your data partition. Open your Windows User folder, right-click your 'Documents' folder and choose 'Properties > Location tab'. Click 'Move' and select the 'Documents' folder on your data partition. Click 'OK', then enable it to move all your existing data to the data partition. Repeat for your other folders.

Once done, boot back into Ubuntu. Your files should now be easily accessible from the data partition, enabling you to make changes to them using both Windows and Ubuntu

programs. To make the change complete, follow the step-by-step guide above to point your Ubuntu user folders to the same location. We recommend following the advice given and leaving the Ubuntu desktop folder alone. Once done, test that the tweaks work by opening *Nautilus* and clicking the 'Documents' shortcut in the left-hand pane, which should now point to your data partition.

To access your network, open a folder window and you'll see network options on the left of the screen. Click 'Browse Network' to find other computers – there's a handy Windows

“Ubuntu can detect both FAT32 and NTFS, so accessing your old data shouldn't be an issue”

Network shortcut for accessing Windows PCs. Alternatively, if you know the computer name, click 'Connect to server' and type **smb://comp-name** then click 'Connect'. Have your username and password ready if necessary.

Ubuntu should automatically handle all of your PC's hardware, using generic drivers if necessary, but you can review what's been installed, plus look for additional drivers (specifically for gaming, Wi-Fi or printing) using the Software & Updates app. Open a Search box, type **software** and then click the 'Additional Drivers' shortcut. It will search for any driver updates that may exist and alert you to any alternative or proprietary drivers you may have installed.

If you store data in the cloud, the good news is that Ubuntu supports a wide range of cloud providers – a good thing, seeing as it recently dropped its own Ubuntu One cloud storage service. Some providers are officially supported on Linux, such as Dropbox (www.dropbox.com/install). If you're looking for heaps of free storage, then Google Drive offers 15GB, but as yet there's no official Ubuntu support. A number of third-party offerings do exist, though, and if you're prepared to pay \$15, InSync (<http://insynchq.com>) is the tool of choice. You could even use *OwnCloud* to set up your own storage...





The vitals of Linux

Discover the key tools and techniques you'll need to know.

Now you've familiarised yourself a little with the Unity desktop, it's time to plunge deeper into the Linux ocean and arm yourself with more of the important knowledge you'll need to use it going forward. Before diving in, however, it's a good idea to take a system backup – as you start to learn Ubuntu, you may make mistakes that could lock you out of your system. (See 'Take another backup', below.)

First, let's take a look at the Linux filesystem and how it's organised into folders. First, Linux uses a different filesystem to Windows (NTFS or FAT32) called ext4. This makes your Linux folders invisible to Windows, which can't natively read ext-based drives. That's fine – we've moved your data on to a shared partition that both OSes can see, and it's better that you can't accidentally access your Ubuntu partitions.

Now, open *Nautilus*, where you'll be taken to your **Home** folder. This works in a similar way to your Windows user folder, and normally contains all your personal files, although we've moved key folders across to your data partition so you can

share files with your Windows installation. Shortcuts to these key folders can be found in the left-hand pane.

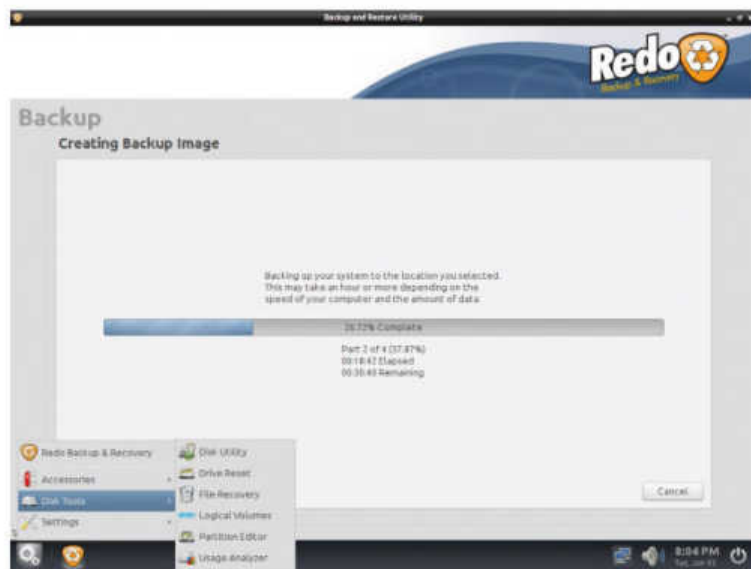
Now click the 'Computer' link under 'Devices' and you'll get a look at a load of other folders, too. Most of these can be safely ignored on a day-to-day basis – the **bin** folder is where your programs are stored, for example – but two worth noting are **media** and **mnt**. These are where shortcuts to any external drives and network folders you've connected to can be found. The key difference between the two folders is that drives mounted in the **/media** folder are mounted on a per-user basis, and don't require elevated access before they are mounted. This means they're not available to the system before you log into your account, which can cause some issues with start-up programs. See <http://bit.ly/MountWinParts> for more information – jump to the 'Configuring /etc/fstab' section for a guide to mounting partitions before login occurs to circumvent this problem.

Key system tools

While Ubuntu will require you to access the command line to perform some complicated tasks, there's a lot you can do without it. We introduced the *Disks* utility in the step-by-step guide (page 17). *Disks* lets you view and manage your drives and their partitions – by selecting one, clicking the 'Settings' button and choosing 'Edit Mount Options' you can make changes to the way partitions are mounted without having to edit the **fstab** file directly. But use with care – if drives don't behave as you expect, flick the 'Automatic Mount Options' switch back to 'On' to restore the status quo.

Many system preferences can be accessed via the System Settings tool – click the 'Settings' button on the menu bar to access it. You'll see three section buttons: 'Personal', 'Hardware' and 'System'. From there you can access a particular preferences tool. Select 'Appearances', for example, to tweak some desktop settings, such as automatically hiding the App Launcher when it's not required and moving an application's menus to its own window rather than the main menu bar.

You can also install a dedicated tweaking tool. We've already seen how the Ubuntu Tweak Tool can be used to move your home folders to another drive (page 17). It also



▶ Take a backup of your dual-boot system using *Redo* for extra security.

Take another backup

Now that Ubuntu is safely installed on your PC, it's time to consider taking a fresh backup of your hard drive. You can, of course, use SystemRescueCD for this purpose (see page 17). If this sounds like too much hassle, you could try *Redo Backup and Recovery* instead. Download the ISO file from SourceForge (<http://bit.ly/Redobackup>) and burn it to disc – in Ubuntu, insert a blank disc and click 'Cancel' when the 'Choose application to launch' window

appears. Locate the ISO file (typically in your Downloads folder), right-click it and choose 'Write to Disc', then click 'Burn'.

When the disc has been created, reboot your PC and choose to boot from the disc – you may need to tweak your UEFI settings before it will work; enabling CSM for storage should be sufficient. Select 'Start Redo Backup' (choose the Safe Mode option if this doesn't work) and click 'Backup' when the main screen appears.

Select your drive, then choose which partitions to back up. Then it's a case of selecting your back-up drive and creating a folder into which your backup will be stored. Give your backup a name and then *Redo Backup* do the rest.

Should the worst happen, recovery is as simple as clicking 'Restore' instead of 'Backup' and following the prompts. You'll also find a 'Settings' button in the bottom-left where a number of useful disk-related tools reside.

All about Grub

When you install Ubuntu alongside Windows, it adds a special boot loader to your system that provides a boot menu for choosing between Ubuntu and Windows on startup. That boot loader is called *Grub 2*, and as you'll have seen, it's designed to wait a short period of time before automatically loading Ubuntu. You can configure how *Grub* behaves – choose a shorter or longer time before it starts loading the default choice, and even change that choice, by editing its configuration file and updating *Grub* via the terminal. Before doing so, make sure you've backed up the system.

Once done, open the terminal, then type **sudo apt-get install gksu** followed by **gksu edit/etc/default/grub &**. This opens the *Grub* configuration file in a *Gedit* window. From here, you need to edit the existing file with whatever changes you wish to implement.

First, the **GRUB_DEFAULT=0** line is what makes Ubuntu the default choice at boot time. To make Windows the default choice, and assuming there are five entries in the boot menu (Ubuntu, Ubuntu advanced options, two memory test options and finally Windows), change this to **GRUB_DEFAULT=4**.

Alternatively, to switch default OS to whichever one was selected previously, change **GRUB_DEFAULT=0** to **GRUB_DEFAULT=saved**, then add a new line immediately below this: **GRUB_SAVEDEFAULT=true**.

You can also change the length of time – in seconds – that *Grub* waits before it boots the default choice. Simply change **GRUB_TIMEOUT=10** accordingly.

When you have done this, click the 'Save' button and close the *Gedit* window. Ignore the warning and type **sudo update-grub** to update the boot loader.

includes a handy Apps tab that makes it easy to locate and install new software. Also take the time to explore the Tweaks section and – in particular – the Unity component where you can do things such as resize the Launcher icons.

Two other tweaking tools that are worth exploring are the *Unity Tweak Tool*, which gives you even more fine control over the desktop, and *Tweak Tool*. Both can be found via the Ubuntu Software Center – choose the *trusty-backports* version of the *Unity Tweak Tool*. Both offer similar point-and-click access to hidden system settings.

Repositories

Although you can download programs from websites, it's far easier to use the centralised Software Center. Ubuntu provides its own software repository, which is a server containing hundreds of Linux programs specially compiled to work with Ubuntu. This means any software you find in the Software Center should work without problems, which makes it a little like having your very own Linux app store. The Center also looks out for program updates, alerting you when they're available and installing them for you.

You're not restricted to one app store – sorry, repository – either; install a program such as *Dropbox* from its own website, for example, and it'll add its own repo to the Center, enabling it to alert you when updates are available.

To manage these repos, open Software Center and select 'Edit > Software Sources'. A list can be found under the 'Other Software' tab, where you can manually add your own.

So far we've not dealt with Terminal, which is Linux's equivalent of the Windows command prompt. While it's technically possible to avoid this, if you want to make the

most of your new OS, then learning to navigate it is essential. Opening Terminal is simple enough – simply press [Ctrl]+[Alt]+[T]. A new window appears and at superficial glance it looks a lot like the command line in Windows. In Ubuntu, you start off in the `~` directory, which basically means your personal home folder. Type **dir** and you'll see a list of folders to confirm this. To change directory, use the **cd** command – to move up a level, type **cd ..**, or type **cd home/nick/Pictures** to jump directly to another folder (note that directory paths are case-sensitive). You can return to your home directory at any time using the **cd ~** command. Other file-management commands include **cp** (copy), **mv** (move), **mkdir** (create folder) and **rm** (delete file).

One key command you'll need to learn is **sudo**. Linux is very tight on security, and in normal day-to-day operations your access is quite restricted. You can't, for example, modify files on your Linux partition outside of your **home** directory. What **sudo** does is give you elevated access, enabling you to manipulate files and perform commands – think of it like right-clicking a program in Windows and choosing 'Run as

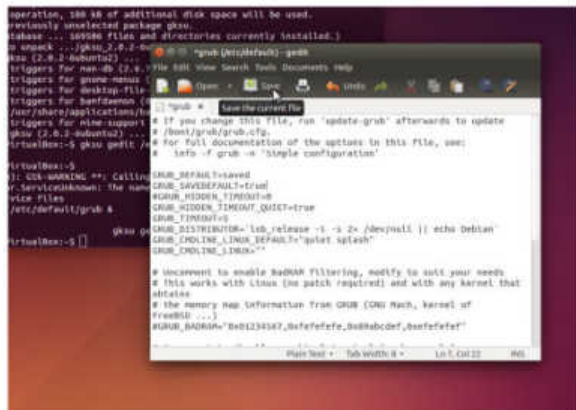
“If you want to make the most of your new OS, learning to navigate Terminal is essential”

administrator'. You can prefix any command with **sudo** to gain this access, but you'll need to provide your password.

You can install software directly from the command line, too, using the **apt-get** command. To install *GIMP* directly from the terminal, for example, you type **sudo apt-get install gimp**. The **apt-get** command can also be used to update all existing software (**apt-get upgrade**), check for broken dependencies (**apt-get check**) and fix problems with missing dependencies (**apt-get -f install**). Over time you can use it to free up disk space by removing Deb packages you've uninstalled (**apt-get autoclean**).

It's even possible to download files using **wget** – simply type **wget** followed by the full URL of the file you wish to download, which will automatically be saved to your personal downloads folder. To find out more about these and other commands, check out our Terminal core skills tutorials, starting on page 102.

Now you've installed Ubuntu alongside Windows and grasped the basics of Linux, there's no turning back – in fact, we suspect you'll wonder why you didn't try it before.



➤ Configure *Grub* to choose which OS loads by default.



Ditch XP and switch to Linux

Linux can be a bit scary for newbies, so we're on hand to demonstrate how to swap dusty old Windows XP for sparkly Linux Mint.

Do you know one of the 12% of desktop PC users out there (www.netmarketshare.com) who still uses Windows XP? Yes? Then do them a favour – tell them to stop and hand them this article. As of 8 April 2014, Microsoft ended all support for its ageing closed-source operating system. This means every new exploit and security flaw will go unpatched and unfixed, and will remain as a gaping hole in people's digital lives. It really is a bad situation, because it's going to lead to identity theft, an increase in spam and more botnet activity. Of course, it also means that vulnerable people will potentially be exploited. One of the problems is that Windows XP works well for a large number of users, and because of this, they're unwilling to change their habits – irrespective of security concerns.

We're here to try to help shift people away from Windows XP. In our view, many people simply don't care what their underlying system is, just as long as it works and offers the same programs and online access – and what better system could we offer than Linux Mint?

Get started now

We'll show you how you can take an existing Windows XP PC, rejig the partitions, install Linux Mint and retain access to your old Win XP files, while gaining the benefits of a modern Linux OS. You can get all of this by following the article and using the cover disc or with a couple of easy downloads.

On the disc you'll find a 64-bit version of Mint, plus the SystemRescueCD bootable live CD. That means you need a 64-bit compatible processor. See the box below on alternative distros if you need a 32-bit build, because it's likely that you'll want a low-requirement distro as well.

Are you unsure about whether you have 64-bit compatibility? Well, Intel 64-bit support was introduced with its Pentium 4 Prescott back in 2004 and has been mostly across the board for desktop processors since 2005 and its

entire Core 2/iX range. In regards to AMD, it introduced the 64-bit compatible Athlon 64 back in late 2003 and this had filtered through its entire line by 2005. So unless a system is more than nine years old, it should be compatible with 64-bit. Don't worry if yours isn't – all distros offer 32-bit alternatives for non-64-bit systems.

Linux Mint can squeeze into 8GB of HDD space, but more is really recommended. Our guide is going to suggest reducing the Windows XP partition (or a suitable data/file partition) to squeeze in room for the Linux boot partition. For this, there needs to be at least 8GB of free space on that partition. In some circumstances, you'll need to delete any unwanted files before this can happen. It's better if you have a lot more free space, but we'll work with what we can get.

You might be wondering why we're working with an additional partitioning program when Ubuntu-based distros like Linux Mint have one built in. The truth is that while the built-in partition tool works well, it's awkward to use and we're really trying to make things as simple as possible. The *GParted* tool is far slicker, and if people finally plump for Linux – and why wouldn't they? – a partition tool is going to be needed for removing Windows and resizing the root partition. In most cases, that's best done from a boot CD.

If you run into issues, there are plenty of online support sites with answers – often these involve the terminal. What's that you say? The terminal is often portrayed as a black slab of death in circles outside the Linux world. It's nothing more than the Linux version of the command line but is far, far, far more powerful. The terminal allows entire servers to be administered – you can hack the world from here! See the tutorials in section 6 if you want to know more.

And that's it. You'll have Linux up and running. Turn the page to see how easy it is to access your old Windows documents and check out the alternative programs you can use. They're free, easy to install and waiting to be discovered.

Distro divas

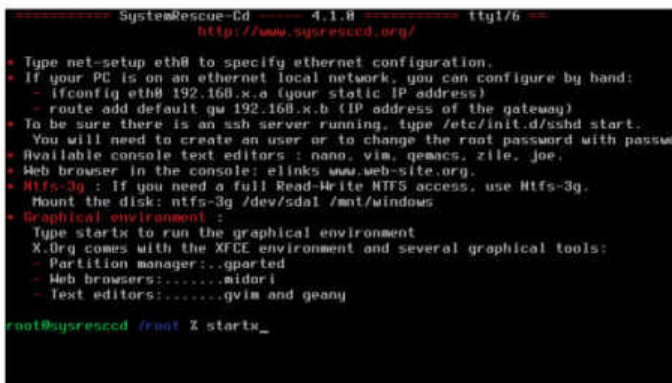
As we've mentioned, we've based this guide on what's available on the accompanying cover disc, but there's no reason you can't substitute these options with your own choices. In some circumstances, this could be for compatibility. A 32-bit lightweight option (such as www.puppylinux.com at 161MB or www.bodhilinux.com at 613MB) may well be the only real choices

for the more ancient XP systems out there. When XP was launched in late 2001, 32-bit was all the rage and remained so, especially on mobile systems, until the mid-2000s.

There are also various releases of both Ubuntu and Linux Mint in 32-bit offerings, and you can grab yourself a download from www.linuxmint.com or www.ubuntu.com.

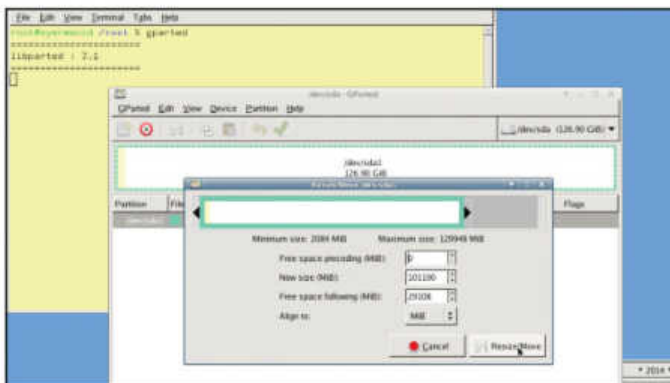
For repartitioning, we're going with the all-powerful SystemRescueCD distro, which is on the cover disc. It might be overkill for a basic repartitioning job, but it's always handy to have around. You might want to consider the more direct *Gparted* Live distro at 175MB from www.gparted.org, which boots directly into a graphical desktop with *Gparted* ready for action.

Bye-bye, Windows XP



1 Gpart it

Boot the system from the cover disc and select 'SystemRescueCD', or create and boot your own. Select the default boot – the alternatives will help solve potential problems you might experience. Select the default keyboard and type **startx** at the final command prompt. This will fire up the default Windows environment.



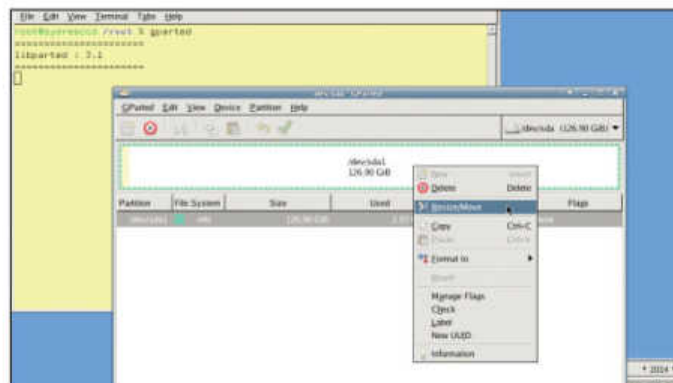
3 Move it

Grab the right-hand handle and resize that partition, you'll want the free space to read at least 8,096MB – ideally this should be a lot more. You'll see over the page how you can still access your Windows XP partition, so you don't have to worry about things there. Click 'Resize/Move' to make the selection, but nothing will happen yet.



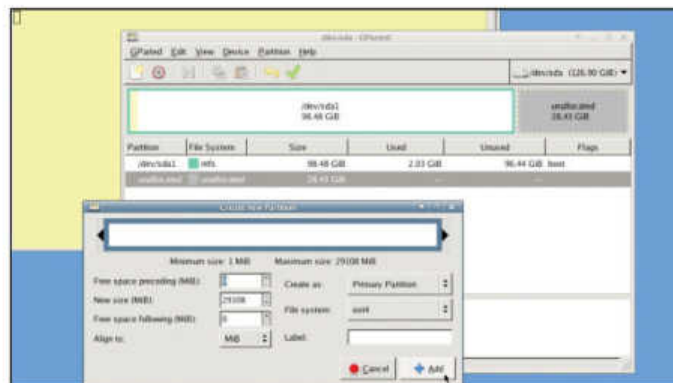
5 Install it

Reboot and switch to Linux Mint. Select 'Install Mint' and you'll see a familiar install system. It'll eventually ask to install Linux Mint alongside Windows XP. Accept and you're on your way. Just like with Windows, you will need a username, network PC name and a decent user password to protect your admin – aka root – access.



2 Resize it

You need to start *GParted*. This is the icon on the bottom-left, or you can just type **gparted** into the open terminal. What you see next depends on the complexity of the host system's partition structure. It could be a single partition, or it could be several. You need to right-click on the one that has the most unused space.



4 Create it

Now you've found some space for your Linux partition, we're ready to go. You won't need to partition or format this space. The installer we're about to follow will identify this as where it should install itself. To apply these changes, click the 'Apply All Operations' button. Depending on the amount of stuff, this can be a quick or slow process.



6 Boot it

It won't take long to install Mint and download the latest updates from the internet. Once in, you should find Linux Mint to be a pretty similar experience to the long outdated XP, except it will be far more secure and – most importantly – it is still supported as an operating system. See over the page for all your replacement Linux programs.

Ditch XP part 2

So you've made the switch from Windows XP to Linux. But how do you make sure you can open all your documents and carry on using your PC?

Just because you have switched from Windows XP to Linux, it doesn't mean that you won't be able to access your existing files, or will have to radically change the way you use your computer. The aim of this tutorial, along with the walkthrough on the previous page, is to make the switch from Windows to Linux as smooth and painless as possible. For many Linux users, the concept of a Microsoft operating system is but a hazy memory. If you are one of those people, these guides are ideal for handing over to friends or family who have yet to make the leap to Linux.

One of the biggest worries for people moving from Windows to Linux is that they won't be able to open their old files, or use the programs that they're used to. Thankfully, this worry is easily allayed, because an increasing number of applications have Linux versions. The beauty of open source software is that if an application isn't available on Linux, there's a very good chance of there being an alternative that will do just as good a job – if not better.

Getting your files

If you followed the guide on the previous pages, you'll have installed Mint while keeping your Windows XP installation and personal files intact. This means that while you are in Mint, you will be able to browse the drive on which Win XP is installed, and access all your files. In Windows XP, these are usually kept in My Documents, My Pictures and so on. Mint, and many other Linux distributions, have similar folders (usually without the 'My' prefix). While it's feasible to copy and paste your documents and files from your Windows XP partition to the equivalent folder in Linux, it's a better idea to leave the files where they are. When you then edit them in Linux, the changes will still appear if you then open them in

Windows. To make life easier, it's worth creating shortcuts to your XP folders so they are easily accessible in Linux. When you open the file browser, you should be able to see the drive or partition on which Windows XP is installed.

Depending on what distribution you're using, the process might be slightly different, but in Linux Mint you can quickly access the file manager by double-clicking the computer icon on the desktop. In the window that opens, you will see all the drives connected to your computer. By double-clicking the drive on which Windows XP is installed, you can browse to your existing files by going to 'Documents and Settings > Username > My Documents', where Username is the name of the account that you use to log in to Windows XP. If you right-click 'My Documents', or on any folder you want to have quick access to, and select 'Make Link', you can then drag the newly made link to your desktop for quick access. You can also add the folder to the left-hand Places bar in the File Explorer menu. To do this, open up the folder, click on 'Bookmarks', then 'Add Bookmark'. The folder will now appear on the left underneath Bookmarks.

As Linux adoption grows, so does the number of Linux ports of popular Windows and Mac programs. *Dropbox*, the popular file synchronisation program, has a Linux client, as does *Skype*. Steam also has a Linux client, and an increasing number of recent games are Linux compatible.

Browse the web

If you use *Google Chrome* or *Mozilla Firefox* to browse the internet, then you'll be able to continue using them in Linux. And the best bit is they will both let you synchronise your bookmarks, passwords and browsing history. If you've been using *Internet Explorer*, you're going to have to make do with

► **Ta-dah! Linux Mint at your service – and isn't it nice?**













a new browser – though that might not be such a bad thing. You can make life easier by exporting your bookmarks from *Internet Explorer* and importing them into *Firefox* in Linux. To do this, open up *Internet Explorer* in Windows and click on the star icon. Click the down arrow to open up the drop-down list, and select 'Import and export', then 'Export to a file'. Select what you want to export and click 'Next' until you get to 'Export' to create the bookmark.htm file.

In Linux, open the *Firefox* browser and click 'Bookmarks' then 'Show all Bookmarks'. A window will open, and you need to click 'Import and Backup' then 'Import Bookmarks from HTML' and select 'bookmark.htm'.

If you were already using *Thunderbird* to get your emails in Windows XP, then getting *Thunderbird* set up in Linux won't be difficult at all. In Windows XP, navigate to the `\Documents and Settings\%username%\Application`

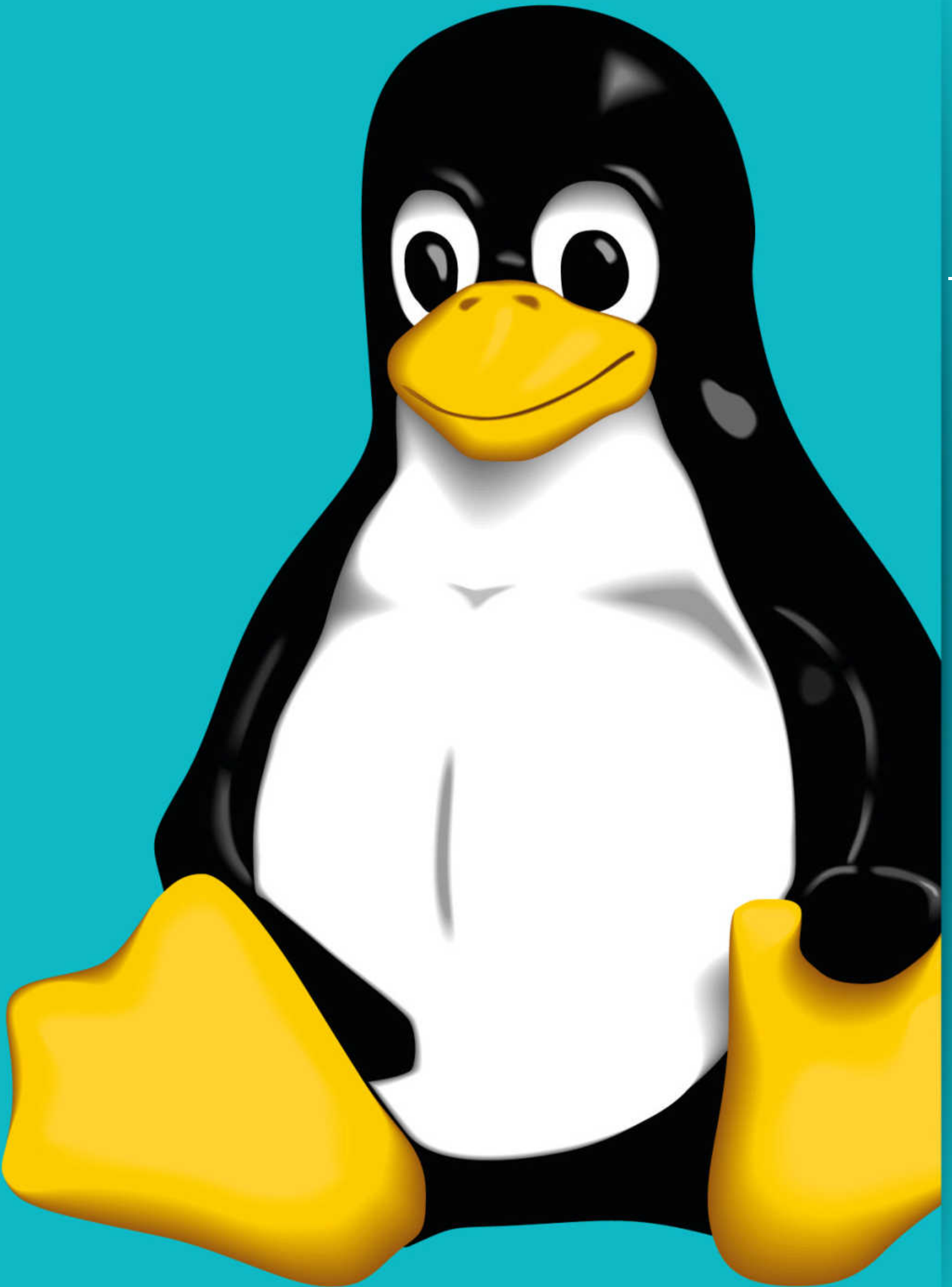
`Data\Thunderbird\Profiles\` folder. In this folder there will be another folder that has a string of random letters and numbers, followed by `.default` as the name. Copy this folder to a USB stick or another hard drive, then load up Linux. Open up the file manager and go to your Home folder (this is sometimes shown as the default view). Press [Ctrl]+[H] on your keyboard to display hidden files and folders, then find the `.thunderbird` folder. Open it, and paste the `.default` folder you saved earlier there. Then, open up the `profiles.ini` file in the `.thunderbird` folder, and where it says `Path=` type in the name of the newly-pasted `.default` folder. When you next load *Thunderbird*, your emails and folders and so on should be waiting for you. If you're using *Outlook* in XP, the easiest way is to install *Thunderbird* on XP, open it up and then go to 'Tools' then 'Import' to import your *Outlook* files. Then, copy the `.default` folder as we described earlier.

Top Linux alternatives

Windows app	Linux replacement	
Microsoft Office	<i>LibreOffice</i> (www.libreoffice.org). It's incredibly easy to replace <i>Microsoft Office</i> in Linux, with <i>LibreOffice</i> being arguably the best choice. It has plenty of tools for word processing, spreadsheets, presentations and more. It's what we use!	
MS Paint	<i>Gimp</i> (www.gimp.org). <i>Gimp</i> is actually much more advanced than <i>MS Paint</i> 's simple image editor, and can even give <i>Photoshop</i> a run for its money. It comes pre-installed on many distros.	
Windows Photo Gallery	<i>Shotwell</i> (www.yorba.org/projects/shotwell). <i>Shotwell</i> is a great replacement for Windows' photo organiser. You can import photos from digital cameras and smartphones, add tags and ratings, and do some light editing to tidy up the shots.	
Windows Movie Maker	<i>Open Movie Editor</i> (www.openmovieeditor.org). For quickly importing and editing home movies, <i>Open Movie Editor</i> is a great alternative to <i>Windows Movie Maker</i> . It's easy to use, but has enough power to make some great-looking clips.	
Windows Media Player	<i>VLC</i> (www.videolan.org). <i>VLC</i> is a very popular alternative to <i>Windows Media Player</i> , and supports lots of different media formats, and can even play Blu-ray discs with a few tweaks.	
iTunes	<i>Amarok</i> (http://amarok.kde.org). If you have an iPad, iPhone or iPod, you might have thought you'd be stuck with <i>iTunes</i> . Thankfully, <i>Amarok</i> does an excellent job of replacing it, has loads of features, and lets you manage songs and files from your iDevice.	
Nero Burning ROM	<i>Brasero</i> (https://wiki.gnome.org/Apps/Brasero#download). Burning files to a rewritable DVD or CD is made incredibly easy with this user-friendly disc burning software.	
WinZip	<i>File Roller</i> (http://fileroller.sourceforge.net). It's likely you already have <i>File Roller</i> installed. It can open or create a wide range of compressed files that are ideal for sending across the internet.	
Norton Anti Virus	<i>Avast! Linux Home Edition</i> (www.avast.com). Linux is far less prone to viruses and malware than Windows, but that's not to say there aren't any or will be in the future, so get some protection.	
Adobe Acrobat Reader	<i>Foxit Reader</i> (www.foxitsoftware.com). The Linux distro you've installed will probably be able to view PDF documents; <i>Foxit Reader</i> is an excellent alternative that's fast, light and full of features.	

Hardware

Build a Linux PC.....	47
Install Linux on a Chromebook.....	56
Build a Steam machine.....	60
Raspberry Pi 2 hands on.....	64



HACKER'S MANUAL 2015

FULLY
REVISED &
UPDATED
EDITION

POWER UP YOUR LINUX SKILLS

- THE KERNEL • NETWORKS
- SERVERS • HARDWARE

180 PAGES OF TUTORIALS

MASTER NEW SKILLS YOU CAN
APPLY TO ANY PROJECT

Future

TGG03 2015

BUILD A LINUX PC

Assembling a PC is straightforward, but choosing the components is less so. We look at the options available.



There is a document that has been floating around the internet for at least fifteen years, called 'What if operating systems were airlines'.

The entry for Linux Airlines states: "When you board the plane, you are given a seat, four bolts, a wrench and a copy of the seat-HOWTO.html". It's an old joke but as Linux users we are still more accustomed to sometimes having to do things for ourselves than users of other operating systems.

Not that this is necessarily a bad thing, as it means we understand our computers better. Still, at least when you want a new PC you can

just go out (or online) and buy one. So why do people build their own? Over the next few pages we will try to answer this question, as well as the more complex questions that arise when you try to do it, such as: how hard is it to do? What are the risks? What about

"Desktop systems are generally very easy to work on – we find LEGO far more taxing."

warranties? Will it save me money? Can I build a computer with no Windows? And many more. There are several reasons why you may want to build your own, not least of which is

the satisfaction of understanding your computer that little bit more, but this information is not only useful if you want to build a new system from scratch. Much of what we cover here will also be of benefit if you are looking to upgrade an existing computer.

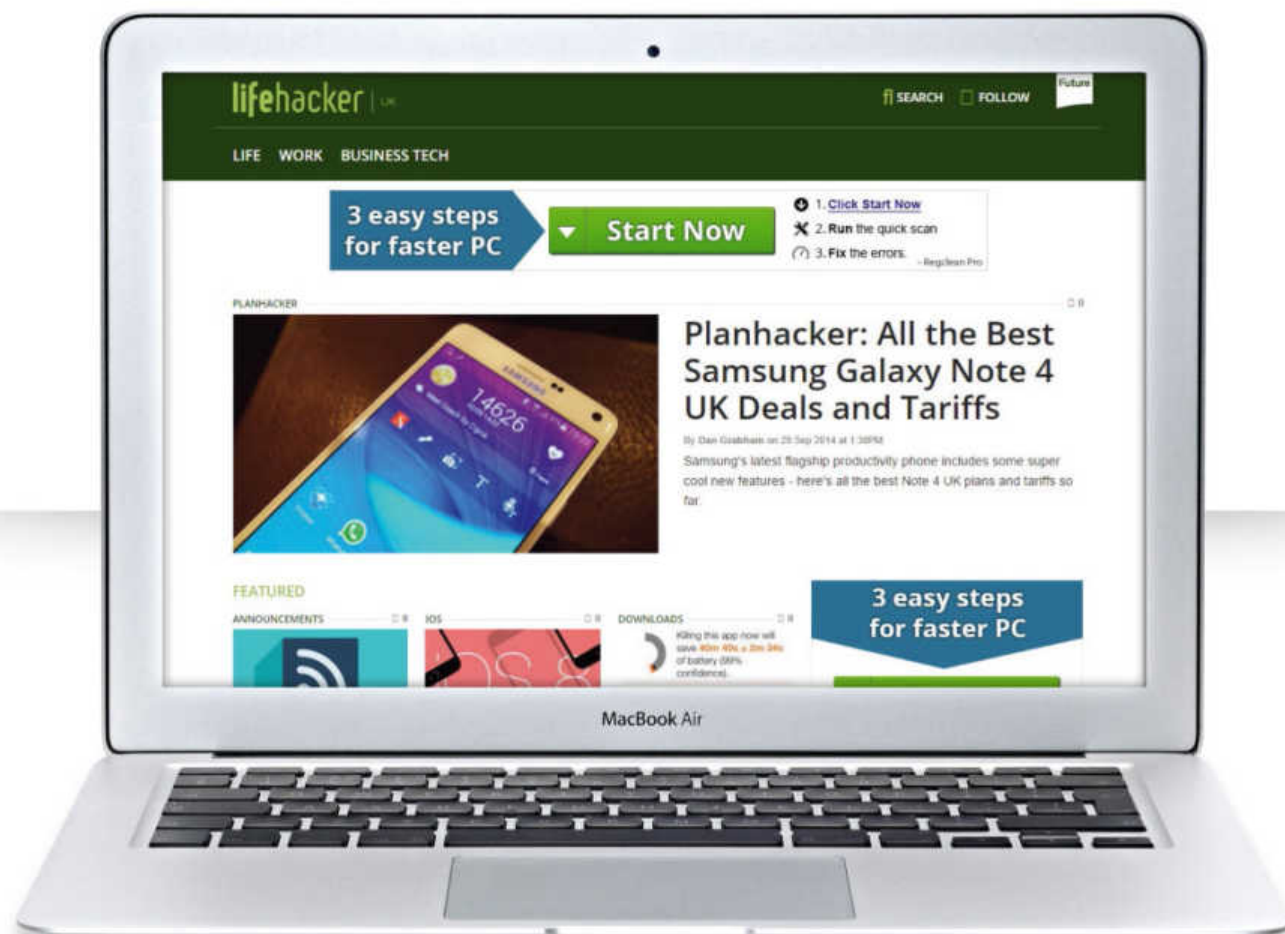
We will be concentrating on desktop systems, which are generally very easy to work on – we find LEGO more taxing.

Laptops are another matter, but many of the points about choosing suitable Linux-

compatible components still apply and we will end with a look at picking a laptop, or any other type of sealed box, such as one of the popular nettop systems. »

lifehacker | UK

Helping you live better & work smarter



LIFEHACKER UK IS THE EXPERT GUIDE FOR ANYONE LOOKING TO GET THINGS DONE

- Thousands of tips to improve your home & workplace
- Get more from your smartphone, tablet & computer
- Be more efficient and increase your productivity

www.lifehacker.co.uk



twitter.com/lifehackeruk



facebook.com/lifehackeruk



» You can buy one of these and get whatever the maker puts inside, or you can assemble it yourself and get exactly what you want.

- » Why build your own computer? You may be able to save money by sourcing the components yourself, but don't bank on this. What you do get to do is pick the exact specification you want – no throwing away parts to upgrade to what you really wanted. You also get to choose the quality of components you use. A pre-built PC may say

“There are still components that are better supported than others.”

it has a 2TB hard drive, but that tells you very little. Most hard drive manufacturers produce several drives of that size, with varying speeds, power consumption and intended usage. You will learn a lot more about components doing it yourself, and while putting together a desktop PC is not difficult, doing so will give you the confidence to dive back inside it when you want to upgrade. You can also upgrade incrementally – once you have a computer you understand, you can increase its hard disk or memory capacity now, then add a faster processor and motherboard next year.

Remember Moore's Law

Moore's Law predicts a doubling of computer capacity every two years. Unless you plan to build a new computer every year, you should allow for your needs increasing during its lifetime. Always allow for spare capacity, if you want 16GB of RAM, buy a motherboard that takes 32GB and half-fill it. Your storage use will

tend to increase too. 4K video is on the way, so build a system with more drive bays and SATA connectors than you need now. Storage and memory are easy to increase, provided you have left yourself the option. Hopefully your processor and motherboard will last you several years.

What do you need?

There are several standard building blocks needed to build a computer. As a minimum, you will need: processor, motherboard, memory, storage drive, graphics card, case and power supply. You may also need a monitor, keyboard and mouse; but you could be building a media centre that uses a remote control and plugs into a TV, or a headless server such as a NAS box.

Over the next few pages we will look more closely at each of these, explaining the choices to be made. The idea is not to tell you which components to choose but to give you the information to make that decision for yourself

based on your particular needs. When you have built your computer, you will then need to install an operating system and software. Naturally, we assume you will install Linux but you

may also need to install Windows for gaming or some other particular software needs, so we will look at how to install the two OSes in perfect (well, almost) harmony.

Choosing the components

When it comes to choosing your components, you need to take into account what you will use the computer for now and what other uses are likely. The usage can dramatically alter the choices you make, eg a gaming system needs fast storage but not a lot of it for the OS and current games you are playing,

so an SSD is ideal. A NAS is the opposite, lots of space needed but speed is not critical, so a slower spinning hard drive would be better.

Linux users have another aspect to consider: compatibility. While the situation is far better than at any time in the past, there are still components that are better supported than others. Let's work through the main components you will need and look at your options.

Processor

The first decision to be made is which processor to use, as this affects your choice of motherboard and then just about everything else. The choice may seem obvious: get the fastest you can afford, but things are never that simple. The fastest processors are also the most expensive, often by a substantial



» Find the price-performance sweet spot for your preferred processor manufacturer.

Linux compatibility

Hardware support in Linux is very good nowadays. Most devices are supported directly in the kernel. There is no need to go trawling manufacturers' websites for drivers. When you are buying components for your own computer, you need to know whether there's support for the hardware before you buy a component. The first step is to determine the exact component in use, which is not as easy as it sounds. The motherboard maker's website may state it has a Gigabit Ethernet port, but not tell you which chipset, so you will have to do some research.

Typing the full product code and the word Linux into your favourite search engine will normally give plenty of hits. It's worth restricting your search to recent posts, there is no point reading complaints about your choice not being supported a year ago when it is now. Look for posts that give details, such as chipset codes and module names and avoid the

rants. Asking in web forums, like our own (www.linuxformat.com/forums) or at www.linuxquestions.org, about the support for a particular device should get you some first-hand experiences, but search the forums yourself first. Your question may have already been answered and no one like questions from people too lazy to do their own research.

Compatibility problems usually arise because the manufacturer gives no help to the kernel driver developers so everything has to be reverse engineered – this commonly applies to wireless adaptors but some sound cards are also affected. Another reason may be that the hardware you are looking at is so new that support has not yet been added to the kernel of your current favourite distro. You may need to look at a newer, more bleeding edge release to get support. This seems to affect things like network adaptors, wired and wireless, the most.

» margin. There's a value for money sweet spot that's usually a couple of steps down from the ultimate. Processor speed is not everything, as memory can have a greater effect on your computer's performance. Going down a step in processor speed and spending the savings on more memory, or an SSD, will usually result in a faster computer. Try leaving a system monitor like *top* running while you use your computer and see how often you use all of your current processor's performance. Unless you are a gamer, or spend your spare time compiling custom kernels, you will be surprised at how rarely your processor usage becomes a limitation.

The other choice to be made is between Intel or AMD hardware. You could try a web search on which is best but be prepared to spend some time searching for real facts among the religious fervour. The truth is that processors are so fast nowadays that for a home system, you would be happy with either. Processor choice affects the choice of motherboard, so it may be that, depending on your needs and budget, you end up choosing a CPU to go with the motherboard you want.

Motherboard

If the processor is the heart of your computer, the motherboard is the central nervous

system. Everything plugs into this board. In the past, that was all a motherboard did: connected everything together. Now they contain a lot of previously separate components built in, such as network interfaces and sound cards. As Intel and AMD use different CPU sockets, choosing your processor immediately reduces the number of motherboards you have to choose from. Now there are some questions you need to ask, such as do I need more than one network interface? How much memory will I want to fit – now and in the future? How many SATA connectors will I need – don't forget to allow for one for your optical drive and possibly an external eSATA connector too. Also, how fast are the SATA interfaces? This is especially import if you are using an SSD. How many USB ports do I need, and what type? While USB 2.0 is fine for a keyboard and mouse, you will want USB 3.0 connectors for storage devices, and possibly a USB-C connector.

Consider the built-in devices of a motherboard you are interested in, such as network interfaces and sound 'cards'. While the motherboards themselves will work with Linux, will the peripheral devices? Sound cards aren't much of an issue these days as most support the Intel HD Audio standard but check anyway as the codecs in use can vary. Wired network devices are also generally well supported, but you may have problems with a very new motherboard that uses a device not yet supported by the current kernel of your chosen distro. As usual, a quick search of your favourite search engine, using the name of the



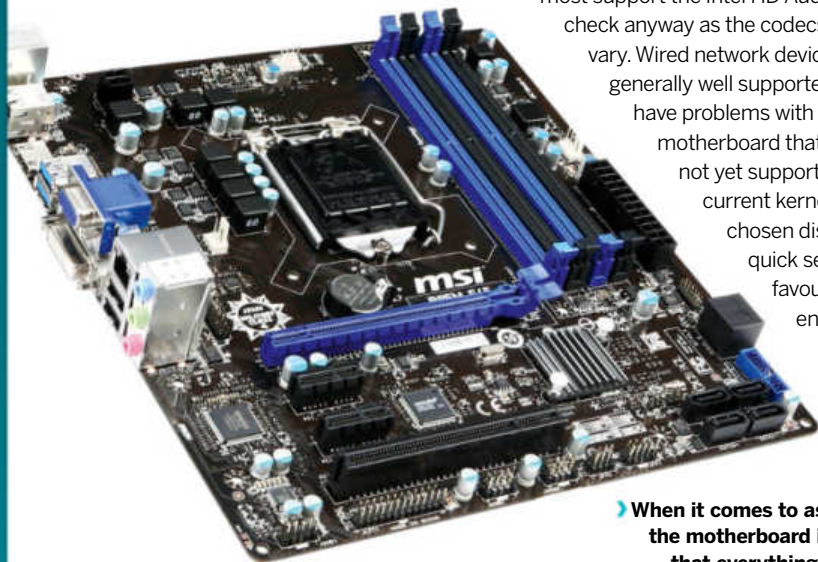
» **Memory may seem unassuming, but it has a greater effect on the performance of your computer than many other components.**

board and the Linux distribution should turn up and potential problems and solutions. Support for onboard devices may not be a critical issue if you have enough PCI slots you can install alternatives and then disable the built in hardware.

Size can be important. If you are building a system in a large tower case, a full-sized ATX motherboard is easier to work with, but for a media centre you may want a smaller form factor that will fit in an attractive case that can go under your TV.

Memory

Memory is one of the simplest ways to boost the performance of a modern computer. It enables more tasks and data to be handled at once and any memory that's left over is used by the Linux kernel to cache disk data which, incidentally, is why your computer shows almost no memory free after it has been running for a while. There are three things to consider when buying memory, the physical layout, the size and the speed. The current memory standard is DDR3 and DDR4. Don't try to fit the older DDR or DDR2 sticks into these slots, they are incompatible. Size is obvious, and it is worth getting the most your budget allows. Check your motherboard's manual for the maximum size for each slot because, although the spec allows for up to 16GB per DIMM, most Intel systems are limited to 8GB per unit. Motherboards usually use a dual-channel architecture for memory,



» **When it comes to assembling a PC, the motherboard is just the thing that everything else plugs into.**

Terminology

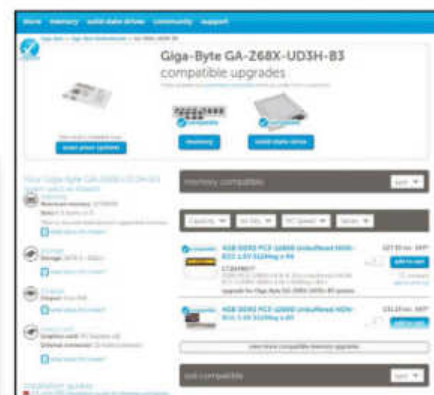
You will encounter plenty of abbreviations and jargon when looking at hardware, here are some of the common terms used:

- » **DDR 2/3/4** Double Data Rate, a technology using in memory chips.
- » **DIMM** Dual Inline Memory Module, this is plug-in memory module.
- » **WAF** Wife Acceptance Factor, seen on male-dominated forums and considered important for hardware that will be in the home.

» **Northbridge/Southbridge** Chips on the motherboard that handle core logic and communication between the CPU and other components.

» **SATA** Serial ATA, the current standard for connecting storage devices to computers.

» **ATX** A motherboard configuration specification, meaning most boards are interchangeable in physical format and standard connectors.



» **Some memory vendors provide databases that are very handy for seeing which memory is best for your motherboard.**

so fit sticks in pairs and check the manual to see which slots go together. This is unnecessary if you are filling all four slots, but unless you are using the maximum size stick, it is better to use two larger sticks. If you want to fit 16GB of RAM, two 8GB sticks give you the option of adding more later, four 4GB sticks does not.

Storage drives

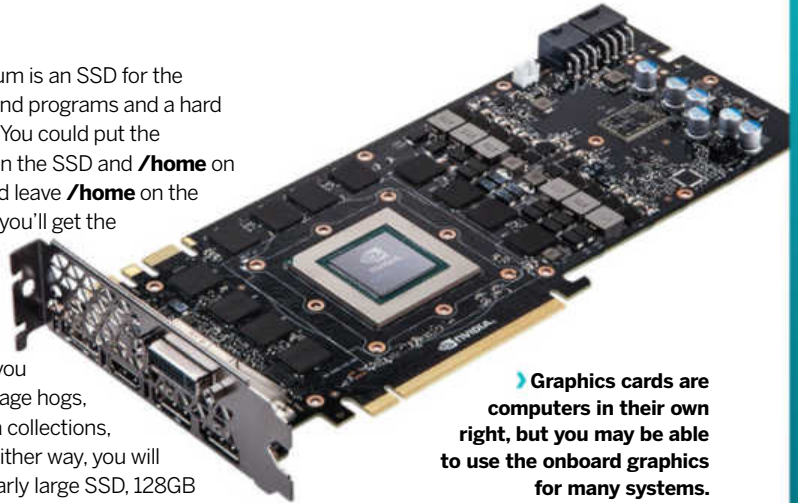
It used to be that the only question to ask about a hard drive was: How big? Now we have to add: How many? And what type? The choice of type is mainly between traditional hard drives with spinning platters and solid state drives (SSDs). (Although there are also hybrid drives that use a spinning disk with an SSD as a cache.) SSDs are much faster than hard drives but cost a lot more per gigabyte, a 256GB SSD costs about the same as a 3TB drive. So an SSD gives much faster booting and program loading but a hard drive holds much more. An SSD also uses less power and is more robust, making it the obvious choice for a laptop. With a desktop, you have the luxury of being able to use more than one



» **SSDs are the obvious choice for a laptop but are idea on a desktop system for installing an OS and the programs you use.**

drive, so the optimum is an SSD for the operating system and programs and a hard drive for your data. You could put the operating system on the SSD and `/home` on the HD, or you could leave `/home` on the SSD, which means you'll get the benefit of the extra speed for things like your web browser and mail caches. Then you can mount the storage hogs, such as your media collections, on the hard drive. Either way, you will not need a particularly large SSD, 128GB should be more than enough, which should mean you can make sure you get a good one. There is quite a variance in the performance of SSDs, so make sure you get one with decent read and write speeds, in the region of 500Mb/s is good. Apart from that, setting up an SSD is pretty much the same as a hard disk.

Another option to consider is using multiple drives in a RAID array. This gives you redundancy, if one drive in the array fails your data is still on the other. This isn't a replacement for taking regular backups but it does protect you against a drive failure. With RAID 1, the simplest configuration, two drives are mirrored. All data is written to both drives but read from one (which can give improved read performance as the data comes from whichever drive seeks to it first). Most distro installers will handle installing to a RAID array, but with RAID 1 you can also install to a single drive and add the second to create the array. RAID is handled by the Linux kernel, do not enable any RAID settings on your motherboard. These are so-called



» **Graphics cards are computers in their own right, but you may be able to use the onboard graphics for many systems.**

fakeRAID and require Windows drivers to work. Just let the installer see that you have two drives and the kernel take care of them.

Graphics card

Most motherboards have reasonable onboard graphics these days, so you may not need a separate card. If you want one, the choice is between Nvidia and AMD, and this is another topic likely to provoke religious flamewars.

Nvidia graphics cards will work as far as booting the computer into a graphical display, but the in-kernel nv drivers are limited.

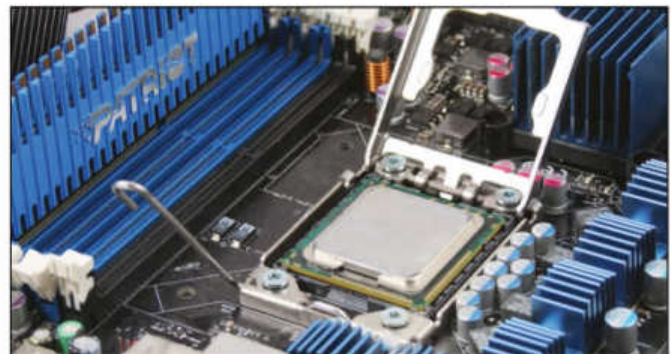
You have a couple of options. The newer open source nouveau drivers work well now – we use them ourselves – and give reasonable performance. If you want the best performance from your graphics card though you will need to install Nvidia's own drivers. As these are closed source they are often not enabled by default so you need to enable the option for restricted or third-party drivers in your distro. You can also download and

Putting it all together



1 Motherboard layout

Find the diagram of the motherboard's layout and connectors in its manual and keep it open at the section. You should refer to it before making each connection to the board. Many of the connectors are simply header pins on the board, making it easy to put something in the wrong way. Murphy's Law is definitely lurking here. RTFM applies to hardware as much as software.



2 Zero Insertion Force

Lift the release lever on the CPU socket, insert the CPU aligning the corner dot on the chip with the mark on the socket. It should drop in with no pressure, then lock it with the lever. Remove the tape from the conductive pad on the heatsink and fit it over the CPU, following the instructions that came with the CPU and motherboard. Connect the fan cable to the motherboard.

» install the drivers directly from **www.nvidia.com**, but this is not recommended as then your distro's package manager cannot track and update them. The situation with AMD graphic cards is similar, the potential hassles of running a binary driver versus the lower performance of the open source drivers. The choice between AMD and Nvidia is almost as religious as the Intel vs AMD CPU choice. Unless you need top notch 3D performance, either will work well for you, and if you need that performance you will have to use the binary drivers whichever way you choose. Currently, Nvidia's proprietary drivers are considered better more reliable, but that could easily change.

Case

The computer's case is often left as an afterthought, but a larger, well made case makes life so much easier. A Cheap case will make cable routing harder and will often have sharp edges that will mean bleeding knuckles and blood stains on your nice new motherboard (you can probably sense the first-hand experience in that statement).

A larger case also provides better airflow for improved cooling. If you are building a media centre that goes in the living room, a high WAF (See *Terminology*, p34) is a major consideration, but pay attention to noise levels. What initially seemed like a whisper can become annoying while watching TV. If you choose to go for a case with a window in the side, bear in mind that you will have to be extra careful when routing cables. A rat's nest of cables is never a good idea, but it's even

worse when it is on show.

Power supply

One of the most overlooked components is the power supply (PSU). Avoid the cheap PSUs that are included with cases (in fact, avoid cases that are cheap enough to include a PSU). A PSU must be reliable and good quality. That may seem an obvious statement that applies to all components, but when a PSU fails it can do so in a way that takes out other components. Having a £100 motherboard or hard drive, or both components, wrecked by a £15 PSU is not a good way of saving money.

Other aspects to consider when choosing a PSU are: that it must supply sufficient power for your needs, now and in the future, that it is

efficient (any PSU worth its salt is rated 80 Plus, but look for Gold, Platinum and Titanium models) and that it's quiet (particularly for a living room computer). There are several websites where you can list the components you are using and get a recommendation of the power supply you need, for example <http://support.asus.com/PowerSupply.aspx?SLanguage=en>. Check that the PSU you choose has connectors you need, most motherboards take a 24+4-pin ATX connector and a separate CPU power lead. Also ensure you have enough drive power connectors of the right type, some PSUs still come with more of the old Molex connectors than the useful SATA power plugs. If you are using a large case with a bottom mounted PSU, check that the leads are long enough to reach the drives at the top, although you can buy SATA power extender leads cheaply enough that deal with the last two points. You'll be pleased to know there are no special considerations to be taken regarding Linux compatibility!

Specialised systems

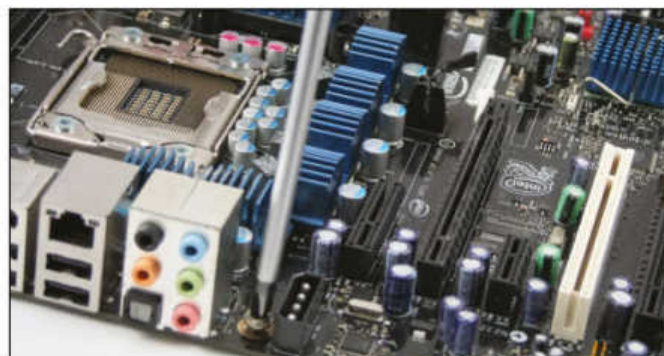
So far we have looked mainly at a general purpose desktop system, but there are some specialist users to consider, such as a home server or a high-performance gaming systems. If you want to build a home server, your requirements will be very different from a desktop or gaming system.

As all data is transferred over the network, a fast SSD is pointless. You could use one for the operating system, but servers are rarely rebooted, so you wouldn't even benefit from the fast boot speed they offer, and they only run a limited number of programs, so loading time is not that important either. Servers generally run headless, so a fancy graphics



3 Memory slots

Insert the memory DIMMS into the sockets on the motherboard. Make sure they are the right way round; slots in the DIMMS should line up with protrusions in the socket and they only go in one way. If you are not using all the slots, consult your motherboard manual for the optimum slots to use and to ensure you take advantage of any dual-channel abilities.



4 Motherboard backplate

Fit the motherboard's I/O plate to the opening in the back of the case and then place the motherboard in the case. There are small posts that it should stand on, you may need to screw them into the case before going ahead and fitting the motherboard. Then secure the motherboard to the posts with the supplied screws – do not overtighten them.

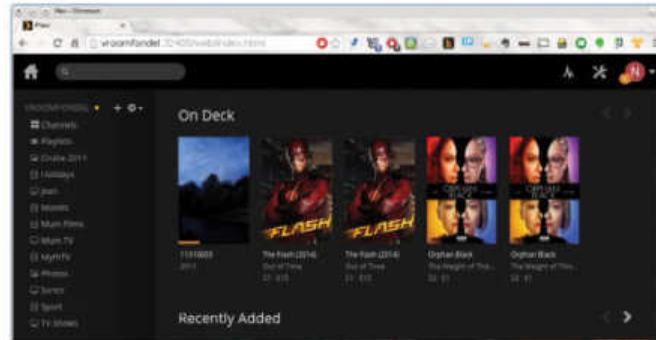
card isn't needed either, you only need a monitor for the installation process and the onboard graphics of most motherboards are more than suitable for that job. What you do need is a decent amount of memory and plenty of storage space. If you are building a file or media server, pick a motherboard with plenty of SATA ports and a case with a similar number of drive bays. However, much storage you add, at some time you will want more. If you are building a web or mail server, storage space is not such an issue, unless you want to serve media files over an internet connection, but plenty of memory helps, especially with *Apache*.

Another type of home server is a media server. This could simply be a repository for your video files or you could be doing the recording on the server too, using *MythTV*'s

“Many of the higher performance graphics cards take up two PCI-e slots.”

backend server software or something like *Tvheadend*. Either way, you will need plenty of space. If you have a server that transcodes video too, maybe something like *Plex* that reformats video to suit the device playing it and the speed of its connection to the server, you will need a decent amount of RAM and a reasonably fast CPU – not a gaming monster but something that's not budget.

As everything works over the network, it goes without saying that it should have a fast network interface, especially if more than one client will be using it at the same time, so



» If you are building a server, like this media server, your needs are different. You need plenty of disk space and a fast network connection (or two). You may also need some CPU horsepower for media transcoding.

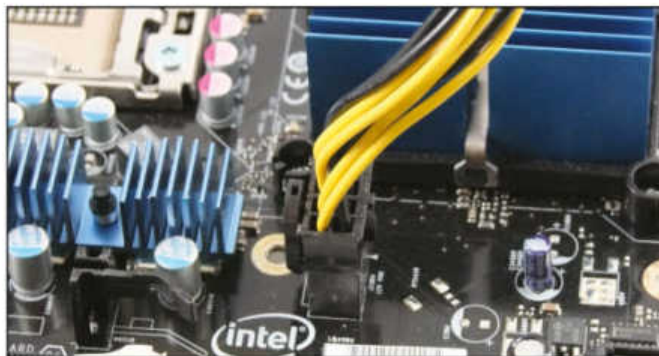
make sure it has a gigabit interface, or budget for a decent PCI-e network card.

High-performance gaming

Gaming, and we mean the graphically intense kind and not gambling, places specific requirements on a computer. Performance is the obvious factor, but not just the

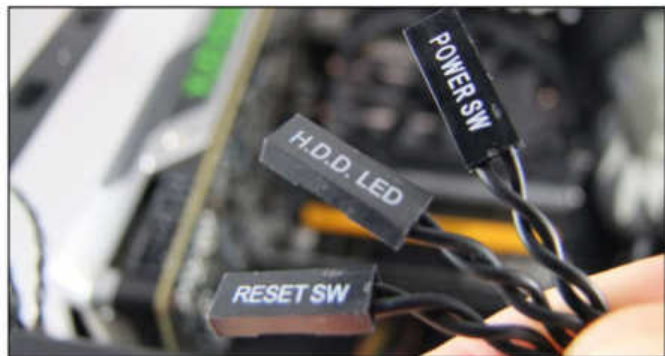
powering it in terms of raw power and physical connectors it can offer. Memory is also important, you want plenty of it and the fastest available. Disk storage space is less important than speed, so an SSD is the preferred choice for storage. Gamers always want more performance and one of the ways of getting it is by overclocking their systems. This is where you run your processor and memory at higher than its rated speed. For general purpose use, opinions on

performance of the CPU. The GPU on the graphics card plays a more important role than the CPU with some games. Clearly you want a fast CPU and graphics card. Many of the higher performance graphics cards take up the space of two PCI-e slots so you need to make sure both your computer's case and motherboard have room for it and that your PSU is capable of



5 Connectors

Fit the power supply and connect the power and CPU cables to the motherboard. This is a good time to connect all fan cables. Add your hard drives or SSDs and your optical drive. They may either screw to the case or in some cases they have screwless fittings. Connect the SATA cables from the drives to the motherboard. Connect their power cables for the PSU.

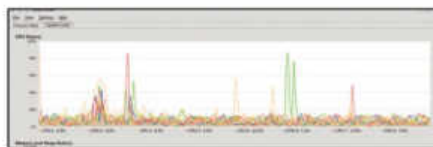


6 Case LEDs

Connect the cables from the case switches and LEDs to the motherboard. These are fiddly connectors but make sure you get them right, the motherboard manual will tell you what goes where. Some of the cables come with a block connector to make the job easier. This is also the time to connect the case's USB ports to the relevant points on the motherboard.

» overclocking are divided; it does give better performance but at the cost of possibly shortened component life and unreliability if you push it too far, plus more heat to get rid of. Some CPUs are locked at a particular speed, so you need to make sure the processor you choose is able to be overclocked. You also need a suitable motherboard. They all offer some control over frequencies and voltages, but some motherboards are intended for overclockers and offer far more control – you'll need to do your research carefully.

To deal with the extra heat, you need a good sized case to maintain plenty of airflow, combined with a decent CPU cooler (the stock cooler supplied with the CPU isn't intended for overclocking) and plenty of fans. A motherboard intended for overclocking may



» **Even with this four-year-old system, the CPU spends a lot of time of doing nothing, but it's nice to have the capacity.**

» **Building your own laptop is not a reasonable proposition, but you can still research the components to make sure you get what you need.**

have more fan connectors and some speed control for them. When choosing cooling fans, go for the largest you can fit. A larger fan shifts more air; fast spinning fans create more noise.

If noise is an issue, consider water cooling. Overclocking used to be a specialist field that required you to buy and assemble various components then spend ages bleeding the system and performing leak tests (it may surprise you, but water and electronic do not mix well). Nowadays pre-built units are available where you bolt the radiator assembly to a suitable point on the case and clip the heat sink in place of the CPU fan. Some kits also include GPU coolers; as mentioned, gaming works them hard too. Water cooling is not only for gamers and overclockers, it can be use on a standard desktop system to reduce the noise. The



loudest noise from such a setup comes from the hard drive stepper motors.

Sealed systems

Assembling a desktop or server computer is a fairly straightforward task, creating your own laptop isn't. While it's possible to buy laptops piecemeal, it's not the usual way of doing things. Still, much of what we have covered in this feature still applies. It is just as important to choose suitable components for a laptop; even more so really because you cannot

Installing an operating system

Once you have built your computer, you will need to install an operating system. We will not look at that here, the distro installers are well enough documented, but there are a couple of points to consider. New motherboards will use UEFI rather than the older BIOS. They may also have Secure Boot enabled. You will almost certainly need to turn this off in the firmware. Press the relevant key at boot to configure the firmware, the motherboard's manual will tell you

which it is (one of the joys of building your own computer is that you get a manual for your motherboard).

You should be able to use UEFI to boot the computer, most distros now work with it, and it's nicer to work with than the old MBR system. If you need to use MBR booting, there will be an option for this in the firmware menu. Some have a UEFI boot option that you can turn off, others have the cryptically named CSM (Compatibility

Support Module) that has to be enabled for MBR booting. If you are building a Linux-only system, you can just boot from your favourite DVD and install. If you want to dual boot with Windows, you should install Windows first and then Linux. The Linux installer will pick up the Windows installation and set up a dual-boot menu, whereas the Windows installer would just trample over the Linux bootloader if you did things the other way round.



7 Graphic card

Fit your graphics card into the PCI-16 slot. Check its documentation to see if it needs an external power feed or whether it gets all the power it needs from the PCI slot. Fit any other PCI cards you may be using at the same time. Many new power supplies (PSU) offer modular cable systems, else adaptors are available if your PSU is missing suitable connectors, but ensure it's powerful enough.



8 Tidy cables, tidy mind

Keep your cables tidy, even if you don't have a window in the case. Untidy cables will disrupt airflow and make life harder the next time you try to add or replace a component. If your case doesn't have clips to hold cables out of the way, small cable ties do a very good job of it. You can also often feed cables behind the motherboard tray or inside the hard drive fixtures.

simply swap them out if you don't like them. So it is a matter of finding a pre-built laptop within your budget that meets your requirements and then checking that everything will do what you want it to. The same applies if you are looking for a nettop to act as a media player.

The same rules apply for choosing a suitable processor, graphic card, sound card and so on (although the use of the word 'card' here is not strictly correct, almost everything is on the motherboard). There are a couple of important differences: laptop hard drives tend to be slower than their desktop counterparts, making the speed advantage of an SSD even greater. Memory upgrade options are limited, so choose a device with plenty to start with and make sure that it has at least one free slot for expansion. Some laptops have only one slot so upgrading memory means throwing away what you have rather than adding to it.

If you have the option, try booting the laptop from a recent live CD, or a live distro on

USB. System Rescue CD with the Alt kernel option gives a very recent kernel with the most up to date hardware support. Run `$ ifconfig -a` to see that both wired and wireless interfaces show up and use `$ aplay` to make sure the sound system works all the way from the given file to the speakers. Wireless interfaces can still be a problem with some laptops, but most have a mini-PCI slot these days, in which case you can try another card.

While the Intel vs AMD CPU war and the Nvidia vs AMD graphics war will continue to rage on, Intel provides excellent support for its components. If your laptop has both Intel graphics and wireless cards, it won't go far wrong and everything should just work with no need to install extra drivers. Now grab your favourite distro and enjoy the speed and freedom of Linux!

Careful now

Building a PC isn't difficult, but some care is needed when working with electronic components. Prepare a large working area, clear of any clutter and with a non-conductive surface. Static electricity can kill electronic components, and it can build up on your body without you noticing, until you pick up a component and zap it. You can avoid this by earthing yourself to discharge and static build up. The simplest way to do this is to touch

a grounded object, such as a central heating radiator or the kitchen sink before touching a component. You can also buy anti-static wrist straps that connect to a grounded point by flexible cable to keep you static free during your build.

The computer you build will not have a warranty, but the individual components will. As long as the fault wasn't caused by damage while fitting them, any reputable vendor will replace them.



8 Testing, testing

Connect a monitor and keyboard and power up the computer. Hold down the relevant key that loads the BIOS/firmware menu and go straight to the system health page to make sure your CPU is running cool enough. Only then should you check that all of your memory and drives are recognised. Some motherboards may need a setting changed to make all memory visible, but your new PC is now built!





Install Linux on your new Chromebook



For those who've bought a Chromebook and miss a full operating system, we'll show you how to get an assortment of Linux distros up and running.

Jargon buster!

apt-get

The program used to install software packages on Debian, Ubuntu and other Linux distros.

Chrome OS is brilliant – for the type of user the Chromebooks are generally aimed at, it does exactly what it needs to do. It is fast and easy to use – what more could you ask for? Well, after a while, you may find yourself missing some of the features associated with more traditional operating systems. But don't fret, because help is at hand, in the form of Crouton.

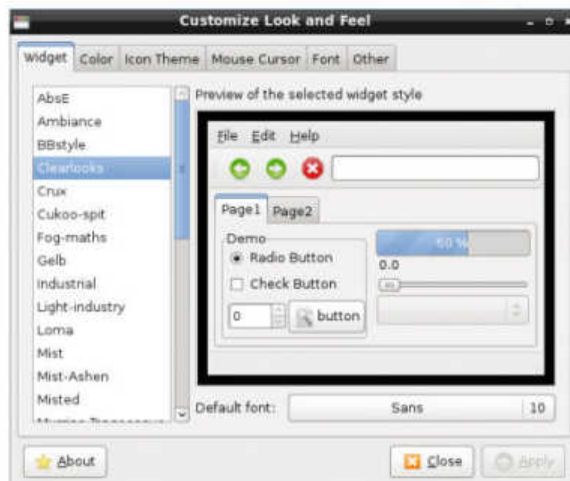
Crouton is a set of programs that set up a **chroot** environment within Chrome OS, from which you can run a Linux OS, with Debian and Ubuntu currently supported. A chroot is not the same as a virtual machine – you are still running on the standard operating system, but within a new environment. This method has several key advantages: for example, it does not touch the existing OS installation,

making reversal easy; it uses the Chrome OS drivers for video, wireless and other devices, so there are no compatibility issues; and it is written by the Chrome OS authors, so it should remain compatible with future updates. The only real disadvantage of using Crouton is that there may be a slight performance hit, but you didn't buy a Chromebook for its blazing speed anyway. Oh, and in case you're interested, the name Crouton is a convoluted acronym (standing for ChRromium Os Universal chroot EnvirONment) that was clearly thought up after the name.

Before you start installing other distros, it's a good idea to create a rescue disk to restore your Chromebook should anything go awry. Even if you're not installing another OS, this is a wise step, especially as it's so simple – all you need is a USB stick or SD card of at least 2GB in capacity. Because of the cloud-based nature of Chrome OS, 2GB is enough, as you only need to back up the operating system – your data and settings are safe on Google's servers. (See the 'Recovery disks' walkthrough on page 59 for details.)

Shell shocked

When you're ready, start downloading Crouton from <http://goo.gl/fd3zc>. This is a script that downloads and installs everything you need. You run it from a shell – yes, Chromebooks come with a shell. Press [Ctrl]+[Alt]+[T] to open the *Crosh* shell in a browser tab. This is a limited shell, but run **shell** to start a proper *Bash* shell. Crouton needs to know which distro you want to install; it calls these releases and selects them with the **-r** option. Then it needs to know the target environment you want to install. A target is a collection of software packages, such as a particular desktop. These two commands will list the options:



➤ **LXDE running on a Chromebook, but Chrome OS is still there.**

Enabling Developer Mode

Using Crouton means putting your Chromebook into Developer Mode, in which you get root access and even a *Bash* shell. This isn't a hack, but a fully supported, if hidden, official option. A warning first: enabling Developer Mode wipes your storage. It doesn't affect your cloud storage but any files stored locally should be uploaded to Google Drive before you proceed. The method of enabling Developer Mode is device-specific – you can find instructions at the Chromium website here: <http://bit.ly/1gDHPGd>.

On the Acer C720 we used for testing, as with most Samsung devices, you turn the device off and then hold down the [Escape] and [Refresh] keys before pressing the power button. This gets you to the recovery screen, then press [Ctrl]+[D] to enable Developer Mode. Other devices have a hardware button for this. Once Developer Mode is enabled, you will see the 'OS verification is OFF' screen each time you turn on – press [Ctrl]+[D] to continue booting, or wait 30 seconds.



➤ Head to www.chromium.org to pick up the Developer Mode switch for your device.

```
sh -e ~/Downloads/crouton -r list
```

```
sh -e ~/Downloads/crouton -t list 2>&1 | more
```

The second command needs to be passed to **more** because it is several screenfuls; hit [Space] to page through them all. Once you've decided the release and target you want, you can run Crouton. To install Ubuntu 13.10 (Saucy Salamander) with the Unity desktop, for example, run:

```
sudo sh -e ~/Downloads/crouton -r saucy -t unity
```

This uses **sudo** because you need root to install the software. You can also specify multiple targets, like this example that installs Debian Wheezy with the LXDE desktop and the *XBMC* media centre:

```
sudo sh -e ~/Downloads/crouton -r wheezy -t lxde,xmbc
```

Starting up

Depending on the target(s) selected and the speed of your internet connection, this could take a while. When it has finished, it tells you the command needed to start your chosen distro in the chroot, such as:

```
sudo startunity
```

Run that command and you will be in a standard Ubuntu desktop. When you've finished, log out in the usual way and you go back to Chrome OS. You can switch between the two by holding [Ctrl]+[Alt]+[Shift] and pressing [Forward] or [Back]. In fact, the Chrome OS navigation keys above the numeric row are treated as [F] keys by Linux, so these are really [Ctrl]+[Alt]+[Shift]+[F1] and [Ctrl]+[Alt]+[Shift]+[F2].

The installation you end up with is not the complete distro as you would get installing it natively, but any extra packages can be installed in the usual way. If using Unity, the *Software Centre* is not installed, so open a terminal in Unity ([Ctrl]+[Alt]+[T]) and run:

```
sudo apt-get update
```

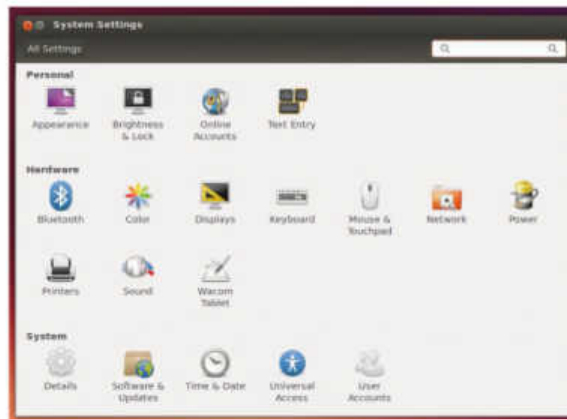
```
sudo apt-get install software-center
```

Now you can install any other packages you need from the GUI. You can also install extra target environments with the **-u** flag. For example, to add the LXDE environment to the Ubuntu chroot we created before, we would run:

```
sudo sh -e ~/Downloads/crouton -r saucy -u -t lxde
```

Adding some privacy

As you may have noticed, enabling Developer Mode gives you root access through **sudo**, without requiring a password. This is slightly less secure for Chrome OS, but your login and files are still protected by your Google login, but it means that all the files in your chroot are readable, even with a passwordless guest login. If this concerns you, it is possible to encrypt the entire chroot by using the **-e** flag for Crouton. This prompts



➤ Unity is perfect for running everything in full-screen.

Jargon buster!

chroot
A directory into which a program is locked. It can't see

for a password and uses that to encrypt the entire chroot directory, meaning you can neither read nor run the chroot without the password. For example:

```
sudo sh -e ~/Downloads/crouton -e -r wheezy -t xfce
```

There are lots of distribution releases and targets to choose from; you could install them all at once but that would get pretty bloated, so how do you try them all out? The answer is that you can have as many chroots as you have space for.

If you plan to do this, you may find it easier to use Crouton's **-n** option to give each chroot a name, otherwise they are simply names after the release. Naming is important when installing multiple releases, because the name is needed when running the start-up commands, otherwise Crouton just loads the first release in which it finds the target you gave. Adding **-n**, like this, lets you ensure the correct release is loaded:

```
sudo startunity -n saucy
```

Crouton also installs a couple of useful tools, particularly *edit-chroot*. This can be used to back up a chroot.

```
sudo edit-chroot -b saucy
```

creates a back-up file in **~/Downloads**, which you can restore with the following:

```
sudo edit-chroot -r ~/Downloads/backup-file.tar.gz
```

Copy this somewhere safe. Even if you do a full reset/recovery, you can still restore it by downloading Crouton again and running:

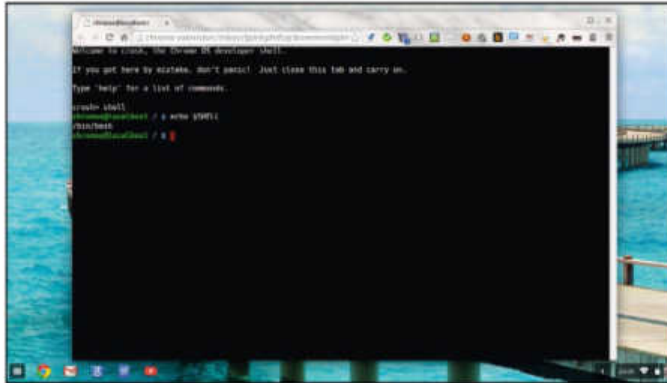
```
sudo sh -e ~/Downloads/crouton -f backup-file.tar.gz
```

You can also use *delete-chroot* to delete a chroot, which you could have worked out for yourself, or you can simply delete the directory holding it from **/usr/local/chroots** to go back to a vanilla Chrome OS. Assuming, of course, that you'd want to do that. Follow the steps over the page...

Quick tip

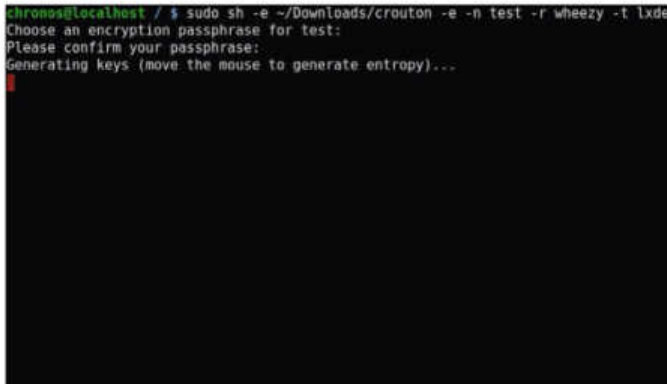
When trying multiple distros or targets, clean out any you have finished with. At several GB each, your storage will soon disappear.

Install a release



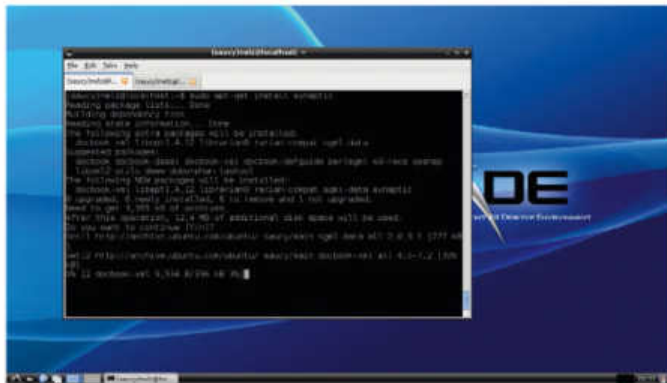
1 Open a shell

Open a terminal window by pressing [Ctrl]+[Alt]+[T]. This will be a basic *Crosh* shell in a browser tab, which has a limited set of commands – you can see them by typing **list**. One of the commands is **shell**, which gives you a full *Bash* shell, like other distros. It's true – Chrome OS has a proper Linux OS behind the scenes.



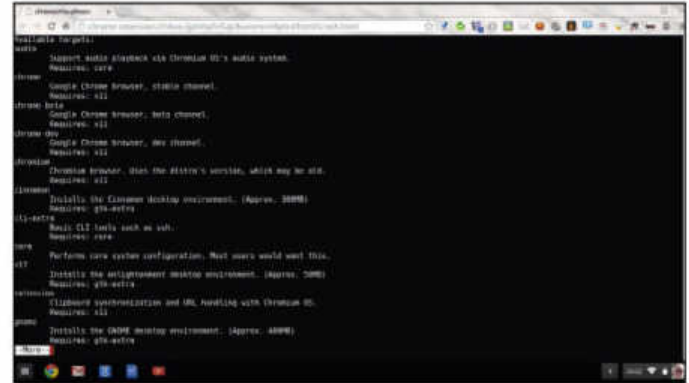
3 Encrypt your files

Adding **-e** to Crouton's command line (this is not the same as the **-e** that follows **sh**) causes your chroot to be stored in an encrypted directory. Choose a decent passphrase – this is all that is protecting your files, but remember that most of your data will probably be saved in the cloud, because Chromebooks have very little storage.



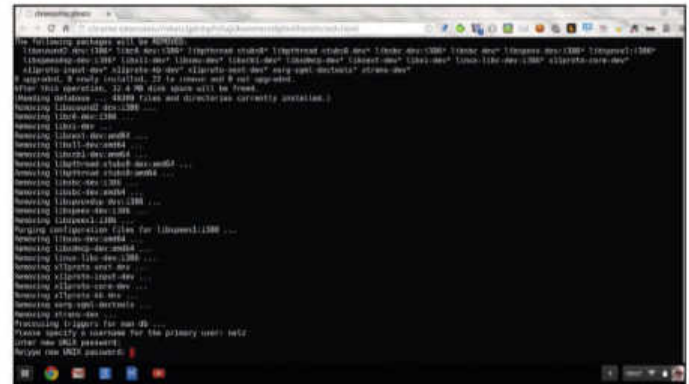
5 Add a package manager

The default targets include only the command line package manager, **apt-get**. For most people, the first step is to open a terminal and use it to install a more friendly option, such as *software-center* for Ubuntu or *Synaptic* for Ubuntu or Debian. Run **sudo apt-get update** to make sure you get the current version, then **sudo apt-get synaptic**.



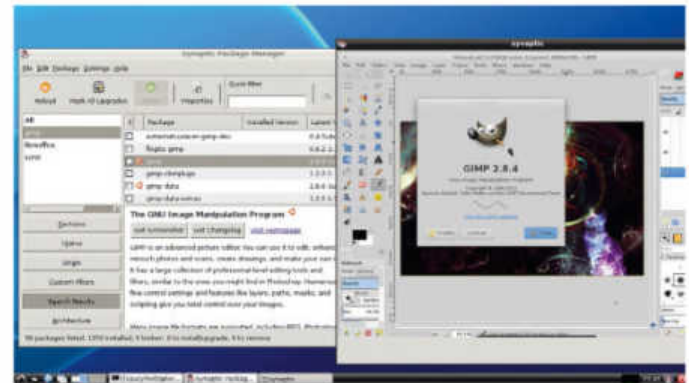
2 Choose a release and target

Running Crouton with **-t list** will show you all the available targets. You'll probably want one of the standard desktop environments. Chromebooks are relatively low-powered, and a lightweight desktop such as LXDE is a good choice, while Unity is better suited to running everything full-screen.



4 Installing the distro

Because Crouton is only an installer, it needs to download the distro release files before installing, so allow time for this. Even with a fast connection, it can take more than 30 minutes to download and install everything if you have chosen large targets – the sizes are shown in the output from **crouton -t list**.



6 Run Synaptic

Once you have *Synaptic* installed, you have easy access to all the software in a distro's repository. Most of the targets are slimmed down, to save on downloads and give a faster installation, but you can install anything you want from here. Either use the 'Search' button or just browse the categories to see what is available.

Recovery disks



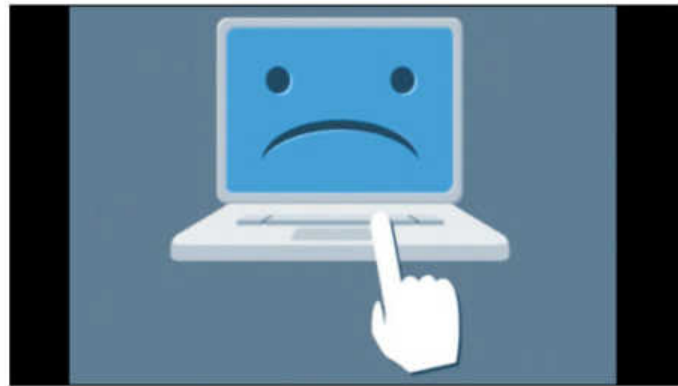
1 Back up to USB

Plug in a USB stick or SD card of at least 2GB capacity, open Chrome and type **chrome://imageburner** into the location bar. Chrome OS downloads and installs the recovery image for your Chromebook. If you have more than one model of Chromebook, run this separately for each one; it gets the correct image for that device.



2 Create the recovery disk

After downloading, the image is written to your USB stick. If you don't create a recovery disk, it's also possible to get this image from another computer and copy it manually, by following the instructions at <http://google.com/chromeos/recovery>, but you have to make sure you get the right image – they are specific to each model.



3 In case of emergency

If you corrupt Chrome OS and get the following scary 'Chrome OS is missing or damaged' message, plug in your recovery medium. You can also force a recovery, if you want to go ahead and restore it anyway, by pressing the hard reset button or key combination, which varies from model to model. Check your Chromebook's documentation for whatever applies.

Crouton: the pros and cons

Comparing a £200 Chromebook to a full laptop may seem unfair – it is more in the netbook price range, and aimed at a similar market – but we'll do that anyway.

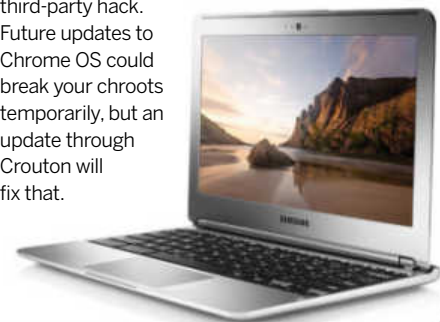
Running Ubuntu or Debian on a Chromebook is just like running it on a proper laptop. The only differences are down to the use of a chroot. This means you have to boot into Chrome OS first and then open a shell to start the chrooted session, but Chromebooks are designed to be suspended rather than shut down, so this isn't necessarily often. Because it uses the hardware through Chrome OS, you need to do things such as set up your network connection in there, but as you can switch between the operating

systems at will, this is not a problem. This isn't dual boot; it's running both systems at once – far more convenient.

The main limitation with this setup is the lack of storage space and dependence on a network connection and cloud services. While Chrome OS handles this transparently, you need to set up some sort of online syncing from your chrooted distro, using services, such as *OwnCloud*, *Spideroak* or *Dropbox*.

There are other ways of installing Linux on a Chromebook but Crouton does it in the least intrusive way, leaving your existing OS untouched (apart from needing to enable Developer Mode). Trying multiple options, and removing them

when done, is also a key benefit of this approach. Not least of its strengths is that Crouton is developed by the Chrome OS authors and isn't a third-party hack. Future updates to Chrome OS could break your chroots temporarily, but an update through Crouton will fix that.





Build your own Steam Machine

Explosive AAA gaming has arrived on Linux with over 1,000 titles available. Building your own dedicated gaming box has never been easier.

Gaming on Linux has been plagued with problems in the past, usually because many developers rush out Linux support – or leave it out altogether – and focus on Windows.

The hard truth is Microsoft's OS is found on the vast majority of gaming PCs (eg Steam's Hardware Survey (Feb 2015) has the Windows user base at 95.68% out of over 125 million active clients), and that's even with the company leaving a trail of broken promises and an even more broken online infrastructure and DRM – try mentioning Games for Windows Live to a PC gamer and see them visibly shudder.

Thankfully, the tide has turned and gaming on Linux is in rude health. Microsoft's desire to create a walled garden with Windows 8 worried Valve, the video game developer behind the much-loved *Half-Life* series, and the company

“It's now easier than ever to game on Linux – we get access to the latest titles.”

behind the Steam service, of course, enough to create a Debian-based distro called SteamOS that is squarely focused on gaming.

Although Valve's embrace of Linux left a lot of us wondering what took them so long, it was

high profile enough to grab the attention of PC gamers who hadn't considered Linux before. With Valve's backing, an increasing number of developers are porting their games to Linux, while hardware manufacturers, particularly

graphics vendors, are making decent strides in supporting Linux through their drivers.

It's now easier than ever to game on Linux – we get access to the latest titles, powerful hardware is supported and we

don't have to struggle getting games working via *Wine* or waste money on a Windows licence. Even better, many PC gamers can even see an impressive improvement in performance just by switching to Linux.

You could, of course, buy a Steam Machine from many reputable manufacturers now (such as Alienware, Asus, Cyberpower, Scan and Zotac etc), but to get yourself a dedicated machine for playing Linux games, we think your best bet is to download and install SteamOS yourself. This distro has been designed from the ground up for gaming, with Steam's Big Picture Mode as the default interface. The interface has been specially built for navigating with a control pad on a big screen, such as a TV, though this means if you want to use your machine for tasks other than gaming then SteamOS isn't for you in its current form.

However, if you want to make the ultimate Linux gaming machine that blows the PS4 and Xbox One consoles out of the water, then head over to <http://bit.ly/BYOSteamOS>.

On this page you'll find two options, the first is to download the default SteamOS beta installation. Although this is probably the most straightforward way of installing SteamOS, it does require a hard drive with a whopping 1TB capacity, which is probably

Agreement. It's worth reading this to understand what SteamOS and Valve's Steam service is.

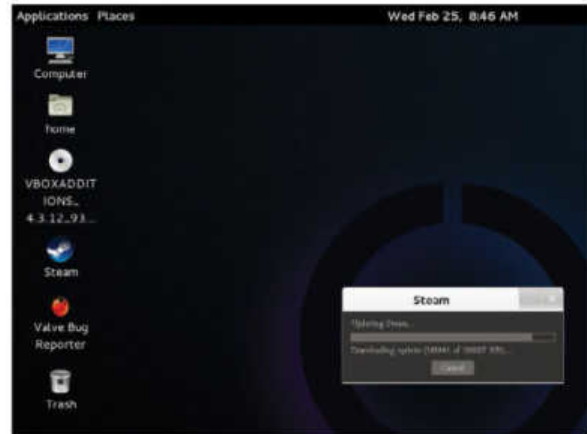
Although SteamOS is Linux-based and uses open source software, it's primarily an interface for Valve's proprietary Steam Store. Proprietary drivers are also used, and although Steam is less obnoxious than some DRM-infused store fronts, you should know what you're getting into before you install it. You will, for instance, be able to access the Gnome desktop that's installed as part of SteamOS to install non-Steam programs and games at least.

Another thing to consider is that the SteamOS is a 1GB download, so if your internet connection isn't the fastest, it's best to have a cup of tea (or four) while you wait. If you agree to the terms of use click the Download SteamOS Beta button to begin.

Once downloaded you'll need to extract the contents of the **SteamOSinstaller.zip** file onto a USB stick. The USB stick needs

to have a capacity of 4GB or higher and will need to be formatted to the FAT32 filesystem.

To format the USB drive to FAT32, insert it and bring up the terminal. Next, type in **df** to bring up a list of the drives installed in your machine. Look carefully at the list to identify your USB stick (capacity is a good indicator). It's almost goes without saying, but what the hell we'll say it anyway, but it's vital you correctly identify your drive before you format



» You can enable a Gnome desktop in the SteamOS settings, which will allow you to run non-Steam programs and games.

it, as going ahead and formatting the wrong one can be devastating.

Once you've identified your USB drive make a note of its path under where it says Filesystem, for example **/dev/sdc1**. You'll need to unmount the drive by using:

```
sudo umount /dev/sdc1
```

where **sdc1** is put the path of your USB drive. Next format the drive with the FAT32 filesystem with:

```
sudo mkfs.vfat -n 'SteamOS' -I /dev/sdc1
```

Once again, where we've written **dev/sdc1**, make sure you put the correct path. Also the **-n 'SteamOS'** part of the code is optional. This just names the partition you've created on the drive for ease of use. If you'd rather not name the partition, feel free to leave this out.

Hopefully, by this point the SteamOS file will have downloaded as a ZIP file. We'll need to unzip the files to the freshly-formatted USB drive. To do this, you'll first need to make sure you have the correct programs installed. As root user type in:

```
apt-get install zip
apt-get install unzip
```

»

The hard stuff for your Steam Machine

When building a machine to play games in the living room you need to consider a few things. For starters, since this is for the living room you'll want it to look good, not take up too much space and run quietly.

For a great looking, yet small PC case we'd suggest going for the BitFenix Phenom Mini-ITX, which can be bought for around £60. Next you'll want a CPU, and although Intel's processors are more expensive than AMD's, they perform better, and will future-proof your Steam machine.

The quad-core Core i5-4570 is a great choice that runs at 3.2GHz and costs around £150. Choosing a case and a CPU narrows down our motherboard options. We've gone for the MSI Z87I AC, which costs

around £50, as it's a Mini-ITX board and compatible with our processor. Even better, the board comes with built-in Wi-Fi so you don't have to trail Ethernet cables through your living room.

Next up you'll want to think about a graphics card. For ease of compatibility we've gone with Nvidia. Some micro-ITX cases have limited space for GPUs, so we've gone for the Asus GeForce GTX 970 DirectCU Mini. This is an excellent and tiny card that will run the latest games with ease. It is, however, a bit pricey at £280, but well worth the money. If you want to save some cash then the slightly older Asus Nvidia GeForce GTX 760 2GB GDDR5 DirectCU II Mini is a great choice and costs a more palatable £187.

You'll also want a cooler (such as the Gelid SlimHero for £25), memory (Crucial Ballistix

Tactical LP DDR3, 8GB for £70 is a good shout), a power supply unit (GX Lite 500W PSU for £41) and a hard drive (any old one will do, we'd recommend 500GB if you're thinking of having lots of games). Hey presto, you've now got an amazing Steam Machine that blows the PS4 and Xbox One out of the water.



» All these lovely components will build a formidable gaming machine.

» Now navigate to the folder where the **SteamOSInstaller.zip** was downloaded (usually Downloads), for example:

```
cd ~/Downloads/
then type in
```

```
unzip SteamOSInstaller.zip -d /path/
```

where **/path/** is enter the path of your USB drive. Next, you'll need to install the USB stick into the machine that you're using for the installation. With the USB stick installed, start up the PC and load up the BIOS. This can usually be done by repeatedly tapping F8, F11, or F12 as soon as your system is turned on.

Once in your BIOS make sure that UEFI support is enabled and select the UEFI entry to boot from.

If you don't mind having the entire hard drive formatted and replaced with SteamOS, select the Automated install option from the menu when it appears. If you have additional disks and partitions that you want to keep, and you want to install SteamOS in a select location choose Expert install.

If you've ever used the Debian installer you'll be pretty familiar with what comes next. First, you'll be asked to choose your language,

location and keyboard layout. The installer will then begin setting up your hardware which will usually take a few minutes. Once done you'll see your hard drives and partitions. This is where you can decide which partitions and drives to use to install SteamOS – useful if you don't want to use all of your hard drive or if you're planning on going the dual-booting route with SteamOS for gaming and another distro for day-to-day tasks.

Select the free space for installing SteamOS – it should be a minimum of 10GB. Select Create a New Partition if you need to

Peripherals

So you've built an amazing, yet compact, Steam Machine and loaded up SteamOS. Now what? You'll want to get some great gaming peripherals for comfy gaming from your sofa.

Valve itself has been working on a dedicated Steam controller with the lofty ambition that it will combine the convenience of a game controller with the precision of a keyboard and mouse setup. It's certainly a tall order and one that Valve appears to have struggled with as the controller has been delayed until late 2015. While we wait for Valve's official controller, which will cost \$50, a number of other

companies offer some great alternatives for controlling SteamOS games. Roccat has built a Sova lapboard especially for SteamOS which offers a small mechanical keyboard and large mouse pad that can rest on your lap. You can also use games controllers from game consoles, such as the Xbox 360 and PS4 as SteamOS does a good job of recognising them as soon as you plug them in. If you're a fan of racing games then the good news is that renowned racers, such as *Project Cars* are coming to Linux. What's not so great is the support for steering wheel controllers.

If you have a Logitech controller you can install the LTWheelConf tool. Full instructions on how to use it can be found on the Steam network (<http://bit.ly/LTWheelConf>).



» The Roccat Sova has been built especially for SteamOS devices.

The 20 best games on Linux

Five best open source games



Strife: Veteran Edition

This is an awesome first person shooter built on the open-source Chocolate Doom engine. Grab the game from <http://bit.ly/StrifeVE>.



Stunt Rally - version 2.5

Race and performing stunts in fantastic environments. This game features 167 tracks, 19 cars and a track editor. Download the game at <http://bit.ly/StuntRally>.



Annex: Conquer the World 4.0

If you enjoy real time strategy games, then this open source game is for you. Download the game from <http://annexconquer.com>.



BYOND: Space Station 13 Remake

This remake of a criminally overlooked classic is completely open source. Download the code from <http://bit.ly/SS13Remake>.



Galaxy Forces: Moon Lander Action!

Hark back to a simpler time for games with this retro-fuelled moon lander shoot-em-up. Download from <http://bit.ly/GalForcesV2>.

Five best AAA games



The Witcher 2: Assassins of Kings

An epic tale of monster-slaying and alchemy, *The Witcher 3* is coming soon, but play this first.



Dying Light

An action survival game presented in first-person. Navigate a dangerous zombie-filled open world to help survivors.



Borderlands 2

This fun and frantic first person shooter makes a post apocalypse world seem like a lot of fun. Play in co-op mode with friends.



Amnesia: The Dark Descent

Games don't come much scarier than this, so if you're after a good horror game then you'll love this.



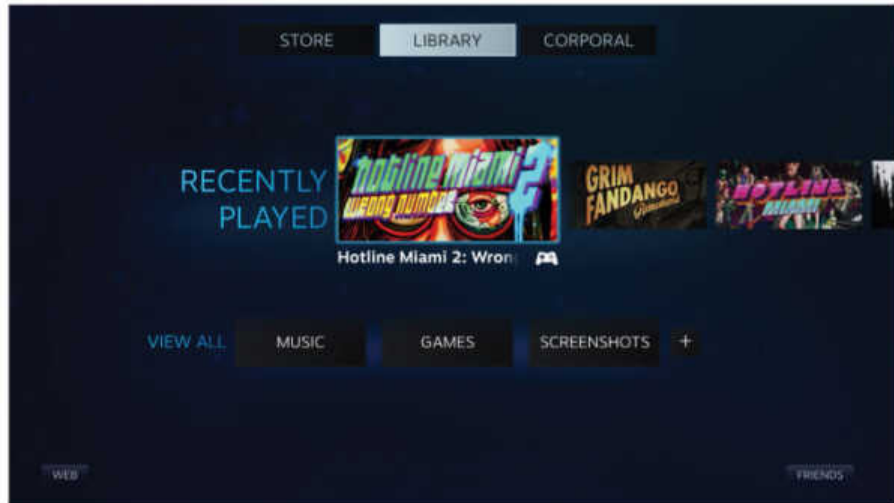
Broken Age

The first graphic adventure by Tim Schafer in sixteen years, funded by a record-breaking Kickstarter – and well worth the wait.

and specify the size. Ensure it's Primary, then click Continue, making sure in the Use as Area it has Ext4 Journaling Filesystem, then select Done setting up the partition.

Click on the free space to create another partition. Give it a size of around 10GB – this will be your swap partition. Make it logical, and create it at the end of the available space when the options appear. When you get to the summary screen, double-click Use as and select Swap Area. Double-click the remaining space, leave the partition size as it is and make sure where it says Mount Point you have it set to **/home**. Select Finish Partitioning and Write Changes to Disk, then select Yes. SteamOS will begin configuring and installing itself. Once done a window will appear called Software Selection asking you if you want to install the Debian desktop environment and standard system utilities. Keep both of these ticked and click Continue. Once done your PC will reboot.

Once your system has rebooted, you'll be given the choice to start SteamOS or start it in Recovery Mode – leave it to start normally and SteamOS will continue the installation. Make sure your machine is connected to the internet, as Steam will be installed. Once that's done your machine will reboot once



» **Big Picture Mode makes launching games on a TV with a games controller quick and easy.**

again. The process might create the rescue partition now, so let it do its thing and select to reboot. You'll then be presented with a Debian login screen. Select SteamOS Desktop and click Return to Steam.

If this doesn't work, open up the Terminal and type **steam**. Accept the terms and click OK. There may be some more downloading to be done, but once that's done you'll be thrown

into Steam's Big Picture Mode where you'll be able to log in to your existing Steam account, or create a new one.

Alternatively, If you don't want to install a new OS for Steam you could install the Steam for Linux client instead on any Debian-based distro by typing in **apt-get install steam** or **aptitude install steam**. You're now ready to enjoy over 1,000 (and counting) titles.

Ten best indie games



Hotline Miami 2: Wrong Number

The sequel to the ultra-violent and maddeningly addictive indie sensation comes with the same thrills and amazing soundtrack, but it's not for the faint hearted or kids.



Supreme League of Patriots

A classic point and click adventure game with very modern sense of humour brings a cast of crazy characters and fiendish puzzles and combines it with a great art style.



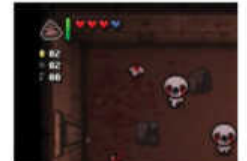
The Fall

The first story in a trilogy, this adventure game excels at world building, story and atmosphere. You play ARID, an artificial intelligence onboard a high-tech combat suit occupied by an unconscious pilot.



Dungeons 2

If you're a fan of Bullfrog's classic game *Dungeon Keeper* then you'll love this spiritual successor, which not only allows you to build devious dungeons to thwart pesky heroes but sees you go topside to attack cities.



The Binding of Isaac: Rebirth

This is a randomly generated action RPG shooter with *Rogue*-like elements. If you don't understand what we just said, all you need to know is that it's a lot of fun.



This War of Mine

A game like no other. You don't play as an all-powerful soldier, but instead a group of civilians just trying to survive in a besieged city.



Chivalry: Medieval Warfare

Besiege castles and raid villages in this fast-paced medieval first person slasher with a focus on PvP.



Papers, Please

Play the role of immigration inspector for a fictional country. Bureaucracy might not seem thrilling, but this manages to make it so.



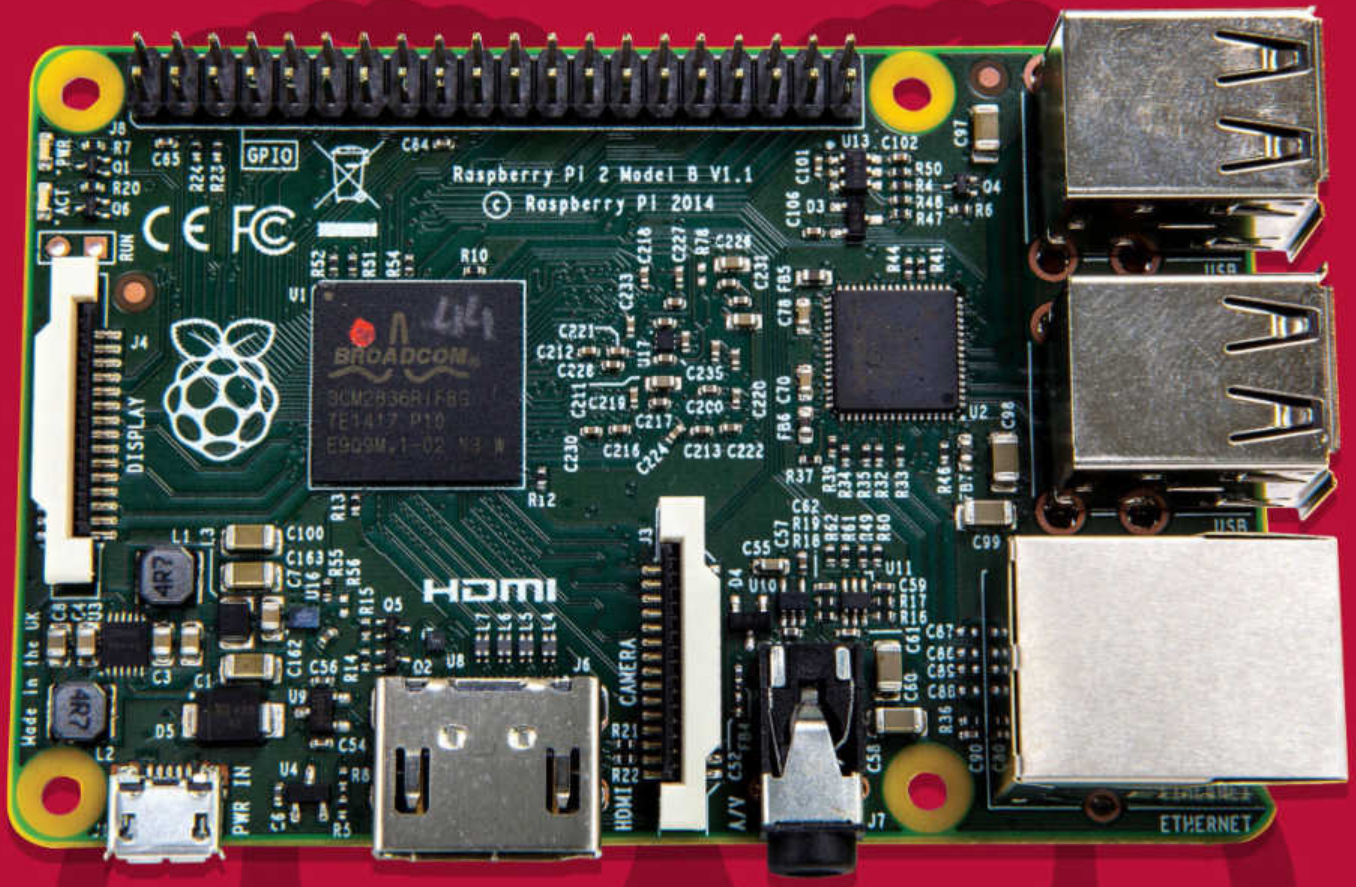
FTL: Faster Than Light

Take your ship and crew on an adventure through a randomly generated galaxy filled with glory and bitter defeat in this spaceship sim.



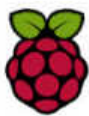
Goat Simulator

Ever wanted to play as a goat? This offers you a chance to live the dream in this completely realistic (not really) simulation.



Hands-on with the Raspberry Pi 2

Our experts get early access to the first batch of Raspberry Pi 2s and take one for a test drive.



The new Raspberry Pi comes with much more power than its predecessors. This is thanks to the improved ARM 7 processor running four cores at 800MHz each and the generous 1GB of RAM. This increase in both CPU and RAM is a massive benefit to projects that require pure CPU grunt, such as OpenCV and Minecraft.

The Raspberry Pi 2 also benefits from the design improvements made for the B+ with more USB ports thanks to the LAN9514 providing four ports over the 9512's two. The B+ also introduced better power management and this, again, is also present in the Raspberry Pi 2. "Power consumption while performing a given task is comparable to that

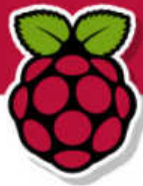
of B+," explains Eben Upton, CEO of Raspberry Pi Trading. "Obviously if you push Pi 2 hard it will consume more power as it's doing more work. Power consumption of B+ under heavy load is roughly the same as the old Model B."

"The Raspberry Pi 2 also benefits from the design improvements made for the B+."

So now that we have the latest Raspberry Pi, let's take it for a test drive! And for this extended tutorial we will be using the latest version of Raspbian available via the Raspberry Pi website (www.raspberrypi.org/downloads) as it comes with the kernel7.img necessary to use the ARM7 CPU.

The easiest method to setup your microSD card is to use NOOBS (New Out Of The Box Software). To use it you will need at least an 8GB microSD card. Download NOOBS as an archive from the Raspberry Pi website and extract the contents to your 8GB microSD card which should be formatted to use a FAT32 filesystem. With NOOBS copied to your microSD card, unmount and remove the card from your computer and insert it into your Raspberry Pi 2, you should hear a gentle click once it is firmly in place.

Connect up your Raspberry Pi to a monitor via the HDMI port and then attach a keyboard and mouse via the USB ports. You will also need to ensure that you have Internet access for your Pi. The easiest method is to use the



Ethernet port. Finally, attach the power supply to the micro USB port. Your Raspberry Pi 2 will now boot for the first time.

On first boot *NOOBS* will ask you which OS to install, in this case we require Raspbian. Select the OS and start the installation, which will take around 10 minutes.

With the install complete the Pi will reboot and start Raspbian for the first time, and the first thing that you will notice is how quick the boot process is now at 17 seconds versus 33 seconds for the B+. For the observant you'll also note that there are now four raspberries at boot up. This denotes that the Raspberry Pi 2 has four cores, a rather nice little Easter egg that harks back to the old boot screens of Linux distributions. Once booted we are immediately presented with the *raspi-config* menu, this is a tool to further configure your Raspberry Pi. At this stage we will simply exit out of the menu and login as normal. The standard login details have not been changed and remain as:

USERNAME: pi

PASSWORD: raspberry

Once logged in you need to type

startx

to load the desktop. You'll see that the desktop is a little different to previous versions of Raspbian, this is due to extensive changes that were made by the Foundation back in December of 2014 and is largely the work of Simon Long, who used to work for Broadcom. In fact, it was Long who hired Eben Upton at Broadcom, but now Simon Long has joined the Raspberry Pi Foundation to work on the user interface and his first project was to develop a new desktop.

The Raspberry Pi Foundation have created a very powerful single board computer, but how can we test that power? It is somewhat fitting that we can calculate Pi to as many decimal places as we wish, so let's say 10,000? To do that we need to install some software, first. Open *LXterminal* and type the following two lines:

sudo apt-get update

sudo apt-get install bc

We've just installed a precision calculator that we can use from the terminal. So now to the test, calculating Pi to 10,000 places and timing the activity.

time echo "scale=10000; 4*a(1)" | bc -l

In our test it took 17 minutes 25.725s to calculate Pi to ten thousand decimal places on our stock Raspberry Pi 2. We repeated the same Pi test on a Raspberry Pi B+ and it took significantly longer at 25 minutes 5.989 seconds to perform the calculation. As you can see straight away that's a very clear indication that the processor of the new Raspberry Pi 2 is easily much more powerful than previous models we've seen.

```
pi@raspberrypi:~$ time echo "scale=10000; 4*a(1)" | bc -l
real    17m25.725s
user    17m25.330s
sys     0m0.020s
pi@raspberrypi:~$
```

► **Computing Pi to 10,000 places is a fitting test for our Raspberry Pi. The command to run the test is run from *LXTerminal*. Here we show the time it took before we overclocked.**

Our quick test, gives us a good idea of how much of a serious performance beast the Raspberry Pi 2 is out of the box, but can we tweak it into more of an animal? Well, earlier we dismissed the *raspi-config* menu but for this next step we need it again. In *LXTerminal*, and type the following:

sudo raspi-config

Turbo charge your Pi

Our first post install configuration is to review the memory split. This is how the memory is divided between the GPU (Graphical Processing Unit) and the main system. On the Raspberry Pi a typical setup would be around 64MB of RAM for the GPU and the remaining RAM would be allocated to the system. This is an optional config tweak but you could just leave it as is. You can easily tinker with these values, and a general rule is that a terminal will not need as much RAM as the full desktop, so for a terminal-only project you can easily get away with 6MB allocated to the GPU. For desktop applications such as *Minecraft* a minimum of 64MB is needed. You will be prompted to reboot your Raspberry Pi, do this and you will be returned to the login prompt.

With the changes made to the memory split now let us return to the *raspi-config* main menu and head to the Overclock menu. Your Raspberry Pi 2 already runs at 800MHz per core, which is an improvement over the original 700MHz single core ARM 11 CPU. We spoke to Eben Upton and Gordon Hollingsworth about the new CPU and they both confirmed that it can be overclocked to around 1.1GHz per



The default web browser for Raspbian, *Midori* has recently been replaced with *Epiphany* which has been optimised for use on the Raspberry Pi. The new browser is available via the latest Raspbian update and works really well on Raspberry Pi 2 and older Pis.

Ubuntu on Pi?

The biggest surprise brought about by the new Raspberry Pi 2 is compatibility with Ubuntu for ARM 7 CPU. Before the original Raspberry Pi was released in early 2012, Ubuntu was often mentioned as a candidate for the Pi, but as Canonical didn't support the ARM 6 architecture, which the ARM 11 CPU used in the original Pi, another distro was needed. Early on we saw Pidora, a fork of Fedora for the Pi, being used to demonstrate the power of the Pi. But Pidora

brought a full desktop experience to hardware that required a lighter distribution. After further investigations Debian, in the form of Raspbian, was seen as a suitable candidate and this remains the official distro and is used for all official tutorials and supporting documentation.

But compatibility with Ubuntu doesn't mean that the Foundation is abandoning Raspbian: "We're not planning an official Ubuntu image," says Eben Upton. "We're going to benchmark

'regular' armhf Debian against Raspbian, and may switch if we see a compelling performance advantage. Our preference is to stick with Raspbian, perhaps with a few key libraries swapped out dynamically, as this will allow us to support Raspberry Pi 2 and Classic from the same image." At the time of writing there are no ready made images for Ubuntu on Pi, but over the coming months there are sure to be many on offer for you to try, including Ubuntu Core.

Quick tip

Watching YouTube videos is now possible thanks to Youtube_dl. Normally YouTube videos are Flash based but there are no Flash packages for Raspbian. When watching a YouTube video in the web browser, Youtube_dl substitutes the Flash element of the web page with an HTML5 compliant video.

» core. We won't be going quite that high, but we will overclock our Raspberry Pi to a stable 900MHz using the Overclock menu. While this is a relatively safe activity it should be noted that going too far with overclocking can severely damage the CPU due to the excessive heat generated by it working harder.

We asked the Raspberry Pi team and it confirmed that the core can reach temperatures of 85 degrees, and when it does it will automatically turn off the Raspberry Pi for protection. For 'extreme' tinkerers who want to push their Raspberry Pi 2 to the limit, now might be the time to invest in a set of heat sinks. If at anytime you wish to return the CPU to it's normal speed, re-enter the raspi-config menu and return the values to the stock 800MHz.

With our configuration changes made, and after a few reboots we have successfully 'turbo charged' our new Raspberry Pi. Let's start the graphical user interface. After logging back in use the *LXTerminal* to type

```
startx
```

to return to the desktop. Now, let's see how the changes have improved our Pi to 10,000 score by repeating the test.

Open *LXTerminal* and repeat the test code which was

```
time echo "scale=10000; 4*a(1)" | bc -l
```

The code will run and in our test it took 15 minutes 28.519 seconds – that's an improvement of two minutes!

The Raspberry Pi Foundation has taken great care to build upon the legacy created by the Raspberry Pi Classic: "Raspberry Pi 2 has been in development for a couple of



» The Hello_Pi demos are a great way to show off your Raspberry Pi 2. You can wrap video around the teapot using any H.264 compliant video.

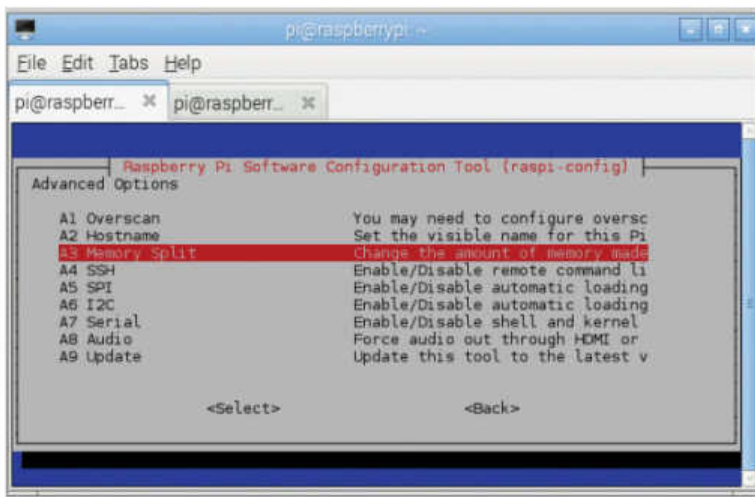
years," say Upton and Hollingsworth, and that includes the time spent developing BCM2836. "The first silicon arrived at the start of last May; there will be a video on the blog of me, James and Dom in the lab at Broadcom at 1am, the day after silicon came back, with the 'video on a teapot' demo running from Linux on a Broadcom "Ray" dev board. The design of the Raspberry Pi 2 board started last August (2014), and we've had samples since October (2014). We went through three iterations of prototypes to get exactly the right performance," says Upton.

Compatibility

The performance is reflected in the choice of CPU for the Raspberry Pi 2. Rather than choose another architecture the Foundation has stuck with an ARM-based CPU that is compatible with the ARM11 found in the earlier Raspberry Pi. The quad-core ARM7 can run software written for the older Raspberry Pi: "Raspbian works out of the box, but requires a new v7 kernel, which will be included in the downloads from our website" said Eben.

As for hardware compatibility, the Raspberry Pi 2 shares the same GPIO as the B+, which means that boards made for the A+ and B+ will also work with the Raspberry Pi 2 and this even includes HAT boards (Hardware Attached on Top), which contain an onboard chip that communicates with the Raspberry Pi to set up the board quickly.

There are some boards, however, that are not compatible with the B+ and the Raspberry Pi 2 due to their size and design. Boards such as PiFace [see **LXF180**] and PiFace Control and Display – that was used to control a camera rig for the Royal Institution Christmas Lectures – cannot be



» The raspi-config advanced menu contains configuration options that enable you to really make the Pi your own.

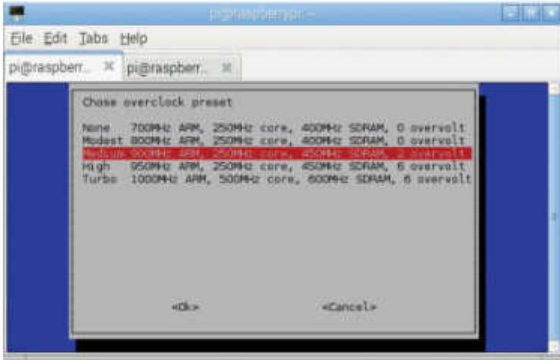
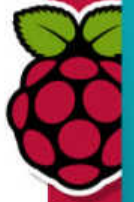
The making of Raspberry Pi 2

The Raspberry Pi Foundation are rather excited about the new Raspberry Pi 2. We spoke to the engineering team and Gordon Hollingsworth about development: "The Raspberry Pi 2 is 100% awesome. It's as close to a typical PC as we wanted when we first started the project." And the amount of effort that's gone into development is impressive: "The team have put the equivalent of 20 years of work into the new Raspberry Pi and it's processor, and that runs at

a cost of between £2-3 million." But there's still a lot of enthusiasm for the older Raspberry Pi. Eben Upton explains: "There are a lot of industrial customers who won't want to switch, and, of course, we still have the Model A+. To give you an idea of the 'stickiness' of an old platform, we sold something like 80,000 Model Bs after the Model B+ launch."

The Foundation also has its Compute Module, which was created to embed the

Raspberry Pi inside industrial applications. We asked Eben if the Compute would be seeing a similar upgrade or not: "We'll do a Compute Module 2 at some point, but probably not in the first half of 2015." And what of the A+? Would there be a similar upgrade for it: "Nothing is currently planned as the A+ price point is quite challenging." No upgrade for now then, but the Raspberry Pi family has grown considerably since 2012 and now features six devices.



» The **raspi-config** overclock menu comes with the required warning that pushing too far will break your Pi.

attached. But the OpenLX SP team behind these boards has released special versions for the B+ and Raspberry Pi 2.

3D graphics test

Every Raspberry Pi comes with the same VideoCore IV GPU (Graphical Processing Unit) that enables the Raspberry Pi to play back high-definition video at 1080p. The new Pi also comes with this GPU, also made by Broadcom just like the BCM2836 powering the new Pi. Did you know that there's a test suite for the GPU?

You can find the test suite by opening *LXTerminal* and typing the following:

```
cd /opt/vc/src/hello_pi/
```

In there you will find a number of directories containing many different demos. But before we can use them we need to build the demos from source, and to make this easier the Foundation have provided an automated build script that will only need to run once. To run the script, in *LXTerminal* type `./rebuild.sh`

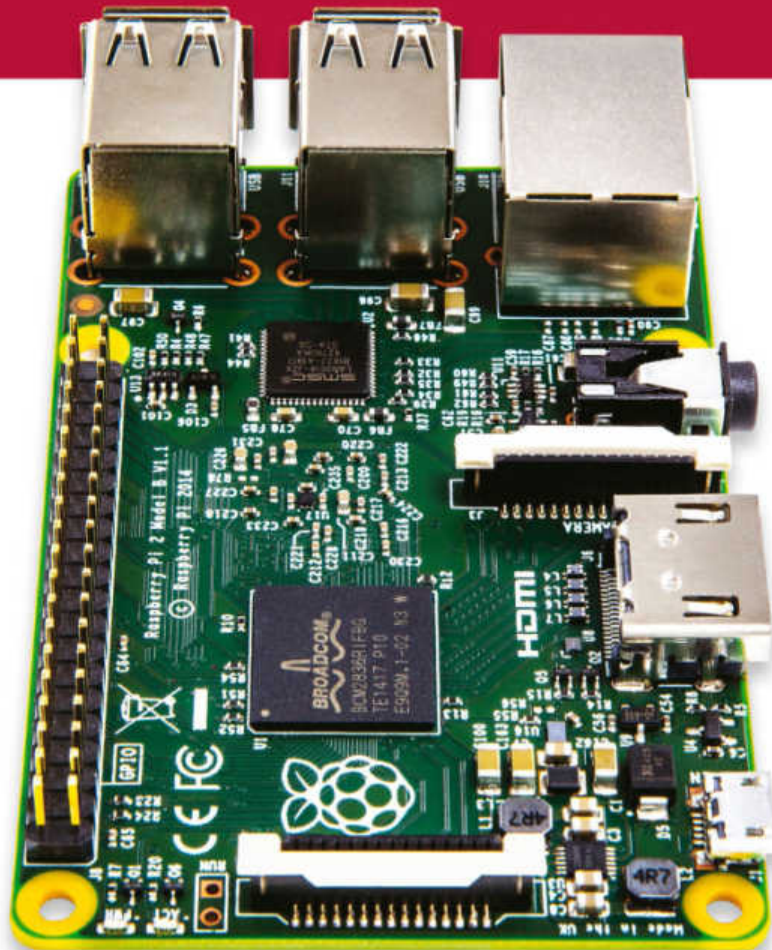
This build script will run the build process for all of the demos so it may take a few minutes, even on our new speedy Raspberry Pi.

Once completed there are a number of demos that you can try out and the first on the list should be `hello_teapot`. To run it, in *LXTerminal* make sure that you are still in the **hello_pi** directory and type:

```
cd hello_teapot
./hello_teapot.bin
```

You will now see a 3D render of a teapot with video that's been directly rendered on to its surface. To exit out of the teapot demo hold Control+C together and you will be returned to *LXTerminal*.

Another demo to try is `hello_triangle2` and to get to that



you will need to go back to the **hello_pi** directory and we can do that by typing.

```
cd ..
```

From **hello_pi** we can change our directory to **hello_triangle2** and run the demo by typing

```
cd hello_triangle2
./hello_triangle2
```

This demo appears to be rather static at first, but try moving the mouse around and you will see two fractals superimposed one over the other moving and reacting to your mouse movements. Apparently, you can also control the fractals to create a perfect circle. To exit out of the `hello_triangle2` demo hold Control+ C together and you'll be returned to *LXTerminal*

So we've taken a look around the new Raspberry Pi 2, seen how it performs out of the box and taken time to supercharge our Pi. On the next page we will interface *Minecraft* with Pibrella to create a pushbutton bomb deployment system!

Quick tip

The Raspberry Pi 2 shares the same dimensions as the B+ but for those of you looking to reuse a B+ case, such as the Pibow, it's worth noting that some surface mount components have been moved. These changes don't affect the overall size of the board but as the Pibow uses layers to build, a new layer will be required for your Pibow.



Updating your Pi

The Raspberry Pi Foundation has released many new updates to the existing Raspbian install and keeping your Raspberry Pi up to date is a really good practice. There are a few handy Terminal commands to help you, such as

```
sudo apt-get update
```

which will update the list of installable software. Using the following

```
sudo apt-get upgrade
```

will check for the latest software, while `sudo apt-get dist-upgrade` is similar to upgrade, but a more intelligent updater that removes old kernels.

If you would like to install the new desktop, and who wouldn't as it is a really great piece of work by Simon Long, type the following three lines into a terminal:

```
sudo apt-get update
```

```
sudo apt-get dist-upgrade
```

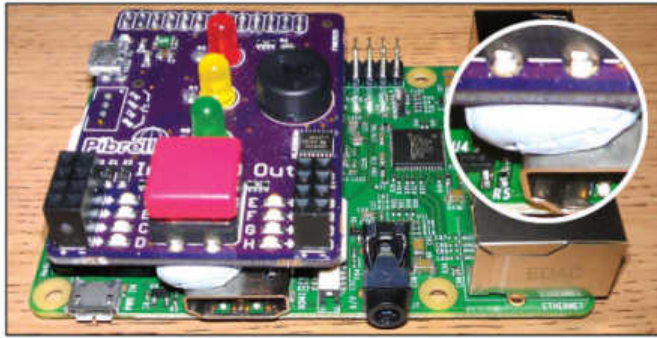
```
sudo apt-get install raspberrypi-ui-mods
```

Once completed, reboot your Raspberry Pi for the changes to fully take effect. Once logged back in you will be greeted with the new and improved interface. Updating is done via the terminal using the very powerful *apt* package management system that's used by all Debian-based distributions.

Link a Pibrella to Minecraft

1 Attach Pibrella to your Raspberry Pi

We'll use the big red button on the Pibrella.com to set off TNT in *Minecraft*. The Pibrella fits over the first 26 GPIO pins. Never attach while the power is on! Use a blob of blu tack or modelling clay to prevent the underside shorting out the HDMI. Now connect the other cables as usual, but insert the power into the micro USB port.

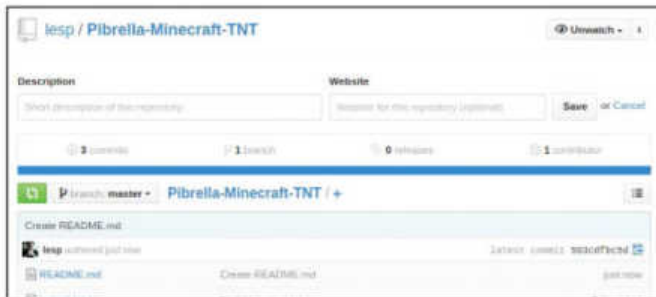


3 Get the code

We've created a GitHub repository that contains the code for this tutorial, visit <https://github.com/lesp/Pibrella-Minecraft-TNT> and download a copy. Next open *LXTerminal* and type

```
sudo idle
```

This opens idle, a Python editor, with elevated privileges enabling us to use Pibrella with Python. Now open the example code.



5 In position

Minecraft uses x,y,z coordinates to know the position of objects in the world. We create a function called **button_changed()**, which locates the player and then creates a cube of TNT at coordinates near to the player. Lastly we set the function to be called when the button is pressed. Keep the window open and open *Minecraft* and create a new world.

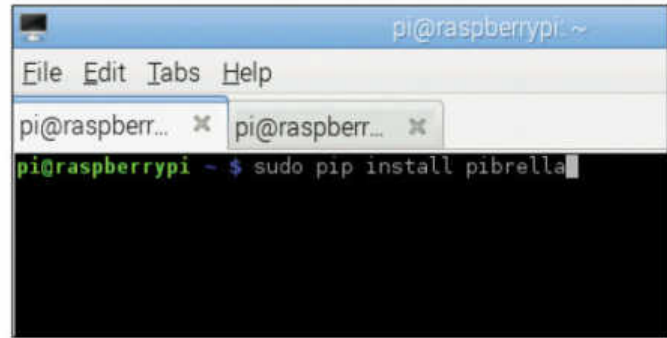


2 Setup Pibrella

With our Pi booted to the desktop, open *LXTerminal* and type:

```
sudo apt-get update
sudo apt-get install python-pip
sudo pip install pibrella
```

This installs the software that we need to use Pibrella with Python.



4 Examining the code

Our code is written in Python 2 as the *Minecraft* module is currently only available for that version. The code is fairly easy to read. Note the line starting with # are comments. The first few lines are imports, they import extra functionality, in the form of Pibrella and *Minecraft* libraries, for our project. After that we use a variable called **mc** to store connection details to *Minecraft*.

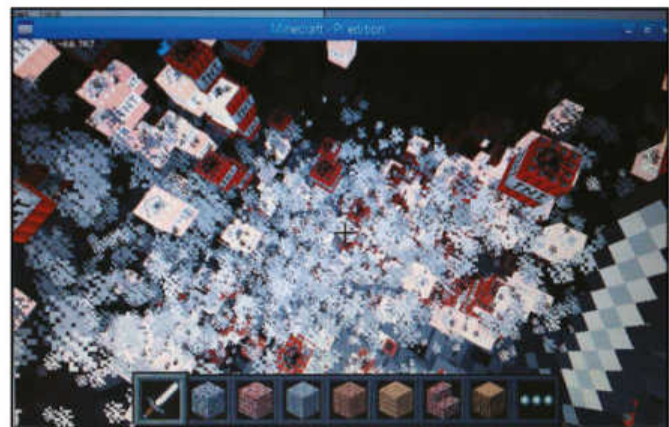
```
import mcpi.minecraft as minecraft
#We import the Minecraft module so

mc = minecraft.Minecraft.create()
#We create a connection between Py

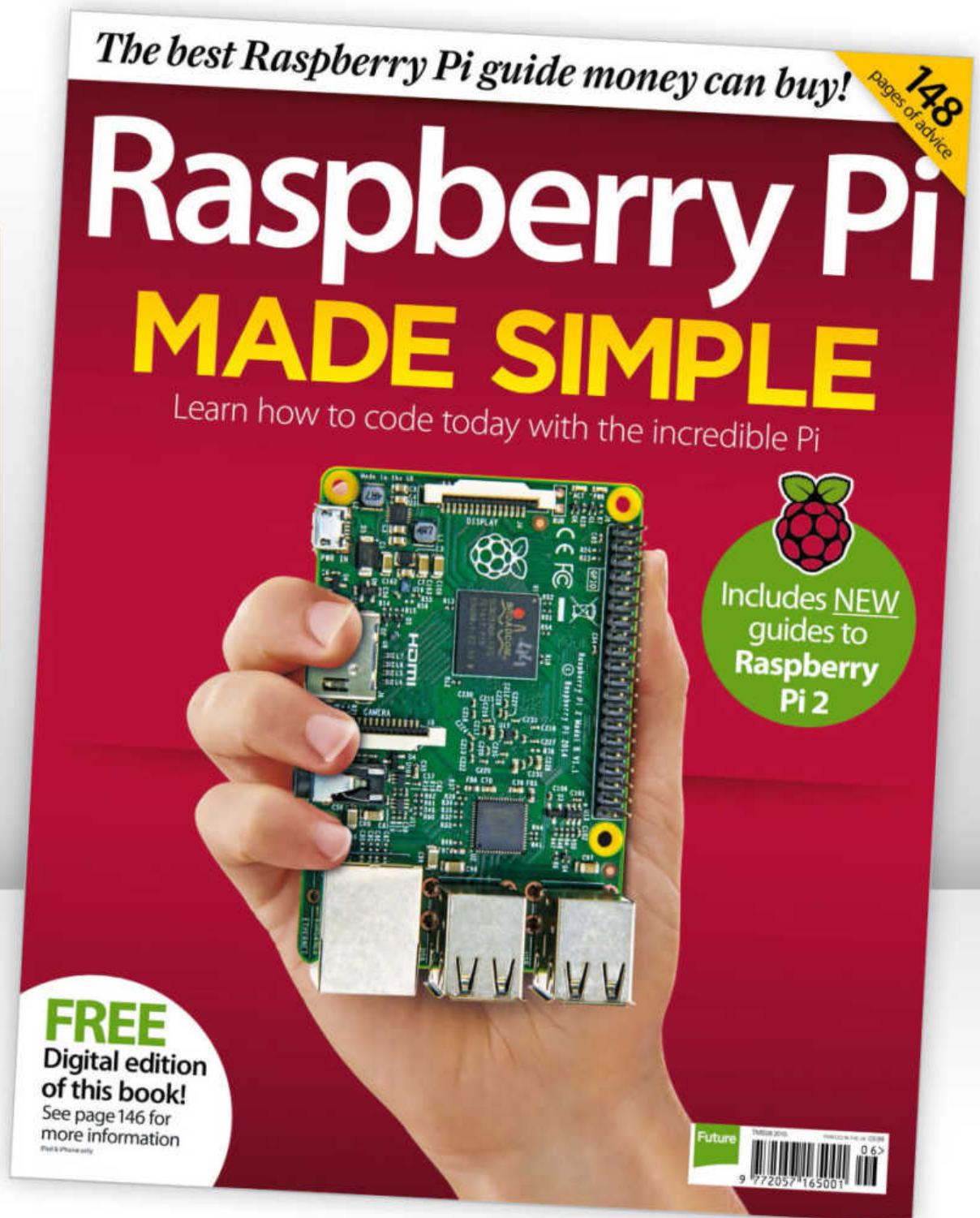
#We define a function and call it
def button_changed(pin):
```

6 Drop the bomb

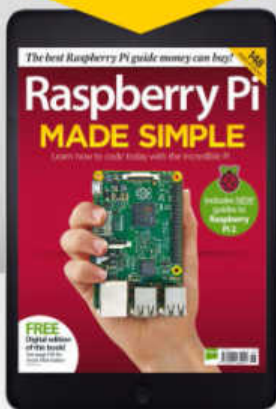
With *Minecraft* ready and our code open, press TAB to release the mouse from *Minecraft* and click on Run > Run Module in idle. The idle shell will come to life and run the code. Switch back to *Minecraft* and go to a nice spot. Press the red Pibrella button to drop the bomb. Hit the TNT with your sword... and then RUN! Note: You can run this the original Pi, but it could crash *Minecraft*.



GET THE MOST FROM YOUR RASPBERRY Pi



**OUT
NOW!**
WITH
FREE
DIGITAL
EDITION

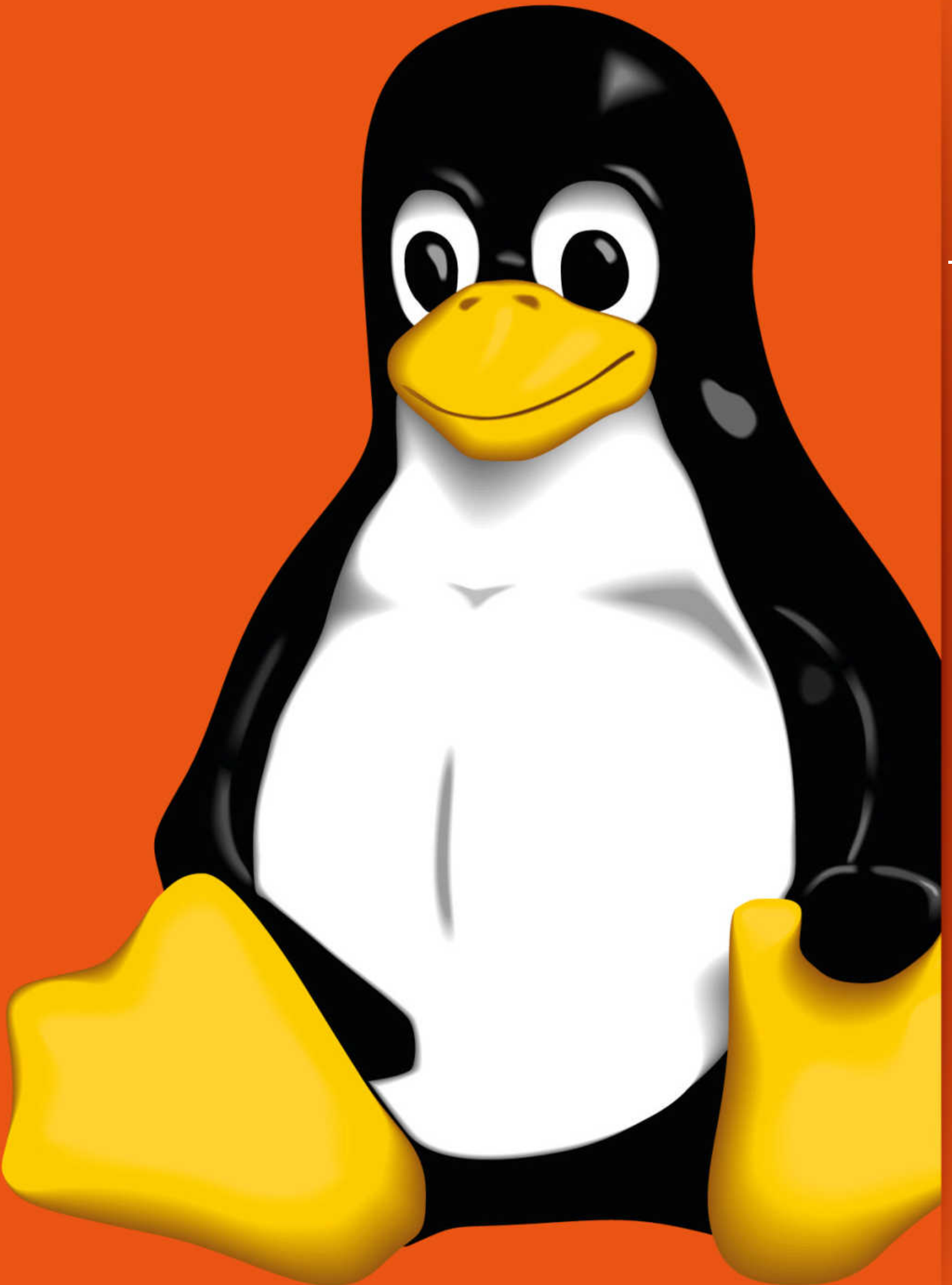


DELIVERED DIRECT TO YOUR DOOR

Order online at www.myfavouritemagazines.co.uk
or find us in your nearest supermarket, newsagent or bookstore!

Software

Pick the right distro.....	73
Low resource distros.....	84
Top 100 Linux tools.....	90
Low resource applications.....	98



SERIOUS ABOUT HARDWARE?



**NOW ON
APPLE
NEWSSTAND
&
GOOGLE PLAY**

**Download the
day they go
on sale in the
UK!**

**MOBO UPGRADES:
HOT USB 3.1 BOARDS**
The latest motherboards
for Intel and AMD CPUs

**THE TRUTH ABOUT
MEMORY BANDWIDTH**
Is capacity really more
important than speed?

**BUILD YOUR OWN
CCTV SETUP!**
Use an old webcam to
help protect your home

PERFORMANCE GEAR & GAMING

PCFormat®

ISSUE 307 • AUGUST 2015

FREE SPEED!

OVERCLOCKING MASTERCLASS

**UNLEASH THE RAW
POWER OF YOUR PC
STEP-BY-STEP GUIDE**

- ✓ Tweak your CPU for top performance
- ✓ Get higher fps in all the best games
- ✓ Hit max speeds with your gear

NO.1 FOR REVIEWS!
Nvidia GTX 980 Ti
Phanteks Evolv ATX
NZXT Kraken X61
and more!

BUILD IT!
**WINDOWS 10
GAMING PC
FOR £559**
PLUS The ultimate
PC buyer's guide

WoW vs SC2 vs Diablo!
**HEROES OF
THE STORM**
Blizzard makes a bid
for the MOBA crown

Future

9 771467 904011

Delivered direct to your door

Order online at www.myfavouritemagazines.co.uk
or find us in your nearest supermarket, newsagent or bookstore!

PICK A DISTRO

If you're diving in to the world of Linux, make sure you pick the right package for you: different distros make a difference.



Your favourite Linux distribution isn't an individual unit in itself. On the inside, it is made up of various apps, libraries, modules and toolkits. On the outside, it's part of a much larger and very vibrant ecosystem that sustains several other distros.

Also part of this larger Linux ecosystem are very active user communities and various support infrastructures that help nourish the distros and other projects. Over the course of their lifetime, the different elements of the Linux ecosystem interact with each other as well as with their environment. They collaborate, exchange ideas and features,

and sometimes swap resources for their mutual benefit as well as for the enhancement of the ecosystem.

The Linux ecosystem fosters the

“The strong survive, thrive and pass on their genetic code to the next gen of derivative distros.”

development of innovative projects and products. However, since the environment can only sustain only so many elements, the distros go through an evolutionary process of their own to weed out the weaklings and ensure the survival of

the fittest. Through this process, the uninteresting, dull and unsustainable projects begin to perish. The strong ones survive, thrive and pass on their genetic code to the next generation of derivative distros.

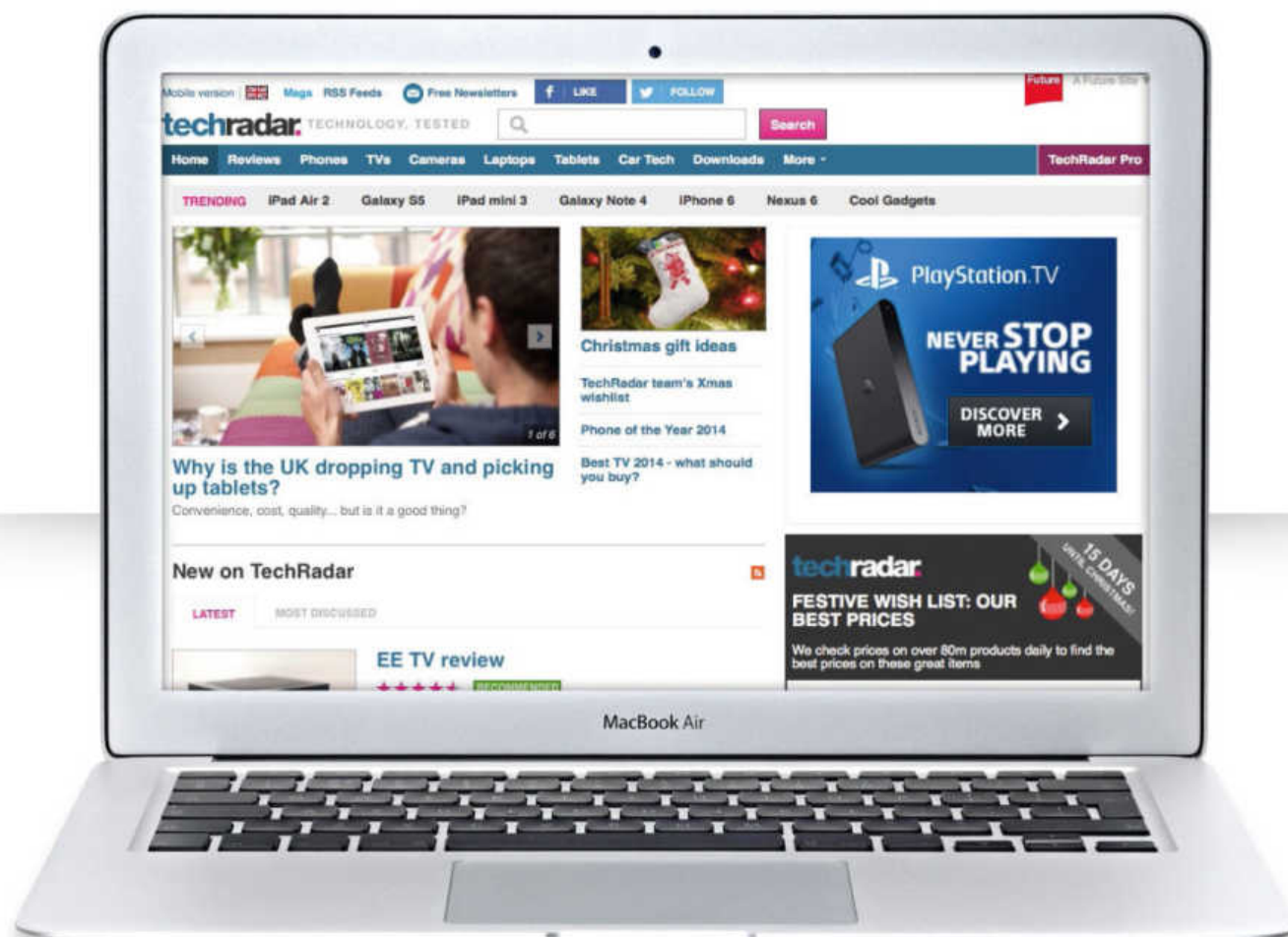
In this feature we will classify the popular distros as per their origins. We will analyse how they've evolved since their inception, and look at the unique traits they have passed on to derivatives that help distinguish them from their peers. We'll also look at the best distro from each distro genus* and then pit them against each other to help you pick the distro that is right for you.



*Genus: A rank used in the biological classification of organisms.

techradar

TECHNOLOGY. TESTED.



VISIT TECHRADAR, THE UK'S LEADING TECH NEWS & REVIEWS WEBSITE

- Up-to-the-minute technology news
- In-depth reviews of the latest gadgets
- Intensive how-to guides for your kit

www.techradar.com

twitter.com/techradar facebook.com/techradar

Genus Debian

Made of free software and evolving.



» **T**he Debian project has played a significant role in the evolution of Linux and, in many ways, is the first real distribution created for the regular computer user. It was announced in August 1993 and had its first public release later that year, although its first stable release wasn't available until 1996. The project was even sponsored by the Free Software Foundation from November 1994 to November 1995.

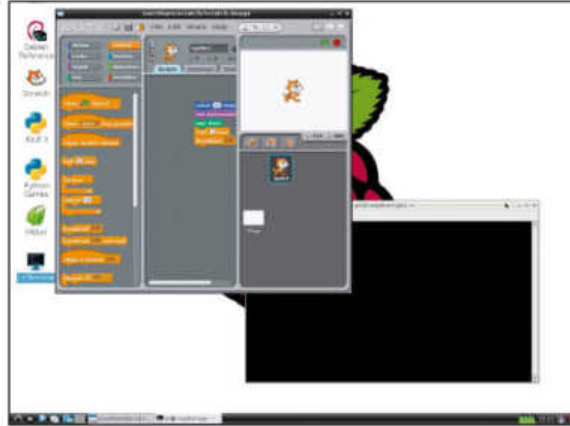
A key motivating factor that led Ian Murdock to create a new distro was the perceived poor maintenance and prevalence of bugs in the Softlanding Linux System (SLS) distro. Besides the software itself, Murdock's release included the Debian Linux Manifesto, which outlined his view for the new project, which prophesied that "distributions are essential to the future of Linux". In the Manifesto, he called for the distro to be maintained openly, in the spirit of Linux and GNU. One of the most significant goals for the distro was to "eliminate the need for the user to locate, download, compile, install and integrate a fairly large number of essential tools to assemble a working Linux system." In order to meet this goal, Debian developers made a significant contribution to the Linux world – the *dpkg* package manager.

This was originally written as a Perl program by Matt

"The Debian Linux Manifesto prophesied that "distributions are essential to the future of Linux."

Welsh, Carl Streeter and Ian Murdock, and the main part of the tool was rewritten by Ian Jackson who became Debian Project Leader in 1998. It really is no surprise then that Debian is one of the most popular choices for derivative projects with over 130 active distros based on Debian (Source: <http://distrowatch.com>), including the likes of Ubuntu and a version of Linux Mint.

The project also provides guidelines to help the derivative distros merge their work back into Debian. In addition to the derivatives there are several 'Pure Blends'; these are subsets of Debian configured to support a particular niche, such as



Debian Edu, Debian Junior and Debian Med. Debian also supports a variety of platforms, including Intel i386 and above, Alpha, ARM, Intel IA-64, Motorola 68k, MIPS, PA-RISC, PowerPC, Sparc and more.

Rules of engagement

Another distinguishing aspect of Debian is that the distro is made entirely of free software. The project uses the Debian

Free Software Guidelines (DFSG) to help determine whether a piece of software can be included. The DFSG is part of the Debian Social Contract which defines the moral agenda of the project.

The project produces three distros: Stable, Testing and Unstable. A Stable

release is available every two years and is made by freezing the Testing release for a few months. Testing is designed to be the preview distro with newer packages and during the freeze any bugs are fixed and extremely buggy packages are removed. All releases are named after characters from the *Toy Story* films (the current Stable release is codenamed Wheezy). All new packages are introduced in the Unstable release (codenamed Sid). This distro is for developers who require the latest packages and libraries. It's not intended to be used on a production machine and those interested must upgrade Debian Testing to get the latest Unstable. »

» Even the most popular distribution for the Raspberry Pi called Raspbian, is based on the Debian Project.

**BEST
OF
BREED**

Linux Mint Debian Edition

The Linux Mint Debian Edition (LMDE) is meant for users who wish to experience the best of Debian (directly rather than via Ubuntu) in an easy to use package. It's based on Debian Testing and is a semi-rolling release, which means it receives periodic updates via Update Packs. These are tested snapshots of Debian Testing to ensure stability and LMDE is binary

compatible with Debian, which means you can switch to Debian Testing or Unstable for more frequent and bleeding-edge updates. However, LMDE isn't compatible with Linux Mint, so you can't use Ubuntu PPAs.

LMDE is designed to offer the same look and functionality of Linux Mint and is available as 32-bit and 64-bit Live DVD images with either the Mate

or Cinnamon desktops. The distro ships with *Firefox*, *Thunderbird*, *VLC* media player and a plethora of other commonly used apps. Adobe Flash plugin and most other multimedia codecs are installed by default. The software repos and the underlying Deb package system makes software installation easy, thanks to tools, such as the Synaptic Package Manager.

Genus Ubuntu

Derivatives, they're coming outta the walls.



» **U**buntu is, in many respects, the first distro to make a serious effort to bring in new users. The distro brought Linux into the mainstream, played a significant part in changing the notion and misconceptions about Linux and was able to successfully pitch itself as a viable OS alternative to Windows and Mac OS.

Ubuntu was started by Mark Shuttleworth. He formed Canonical after selling his security-firm, Thawte, to VeriSign. Shuttleworth was a huge fan of the Debian project. However, there were many things about Debian that didn't fit in with Shuttleworth's vision of an ideal OS. He therefore invited a dozen or so Debian developers he knew and respected to his flat in London in April 2004 and hashed out the groundwork for the Ubuntu project.

The group decided on a bunch of characteristics for the distro. For one, Ubuntu's packages would be based on those from Debian's unstable branch. However, unlike Debian, Ubuntu was to have a predictable cycle with frequent releases. To put the plan into action, it was decided that Ubuntu would release updated versions every six months and each release would receive free support for nine months. The plan was refined in later years and now every fourth release receives long-term support (LTS) for five years.

The group also decided to give emphasis to localisation and accessibility in order to appeal to users across the world. There was also a consensus on concentrating development

efforts on ease of use and user-friendliness of the distro on the desktop. The first release of Ubuntu was in October 2004.

Ubuntu's development is funded by Shuttleworth's UK-based Canonical, which is a privately held computer software company. The company also supports development of other Ubuntu-related projects, for instance, Ubuntu's *Ubiquity* installer is one of the best tools for the job, and one of its distinguishing features is that it gives users the option to install closed source or patented third-party software, such as Fluendo's MP3 codec. Other useful user-centric projects that have tried to change the status quo are the *Ubuntu Software Center* and the recently discontinued Ubuntu One cloud hosting service.

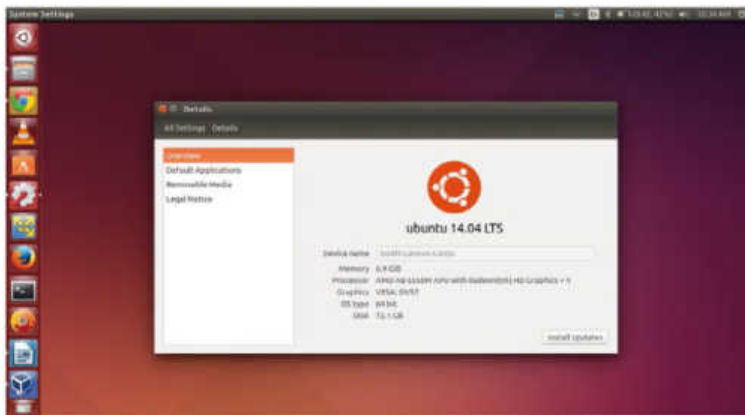
Test by fire

But perhaps no other piece of technology has polarised the Linux community like Ubuntu's Unity desktop interface. The distro first introduced Unity with the Ubuntu Netbook Edition version 10.10. By the time 11.04 rolled off the press, the Netbook Edition had merged into the desktop edition and Unity became the default graphical interface for the Ubuntu distro. However, Shuttleworth has insisted that the Unity desktop plays a crucial role in Ubuntu's multi-device strategy. Unity will help standardise the display on smartphones, tablets, TV and other devices beyond the computer.

Thanks to its malleable nature, the distro has always been very popular with developers who want to create a custom distro for their particular niche. Ubuntu has perhaps seeded more distros than any other, and Ubuntu itself has several officially-supported spins: Kubuntu, Xubuntu, Ubuntu Gnome, Edubuntu and Ubuntu Studio. In addition to the main desktop edition, there's also a server edition that doesn't ship with a graphical desktop.

Ubuntu has helped give Linux mainstream coverage and has several celebrity users, including Cory Doctorow and Stephen Fry. However, pushing the envelope has its drawbacks and the award-winning distro has had its fair share of brickbats. It's still reeling under the Amazon controversy that arose when the distro included search results from the shopping giant in Unity's Dash whenever users searched for stuff on their computer.

» A number of vendors, such as Dell and Lenovo, offer computers pre-installed with Ubuntu.



**BEST
OF
BREED**

Trisquel GNU/Linux

Trisquel GNU/Linux goes to great lengths to do justice to its free software tag. Not only does the distro not include any proprietary software, it also strips out all non-free code from the components it inherits from Ubuntu, such as the kernel. Instead of the stock Ubuntu kernel, Trisquel uses the Linux-libre kernel that doesn't include any binary blobs. Thanks to its

efforts, the distro has been endorsed by the Free Software Foundation.

There are several variants of the distro, the most common ones are the standard Trisquel release, which is available as a 700MB image with the Gnome desktop, and Trisquel mini, which is designed for older hardware and low-power systems, and uses LXDE, the lightweight desktop.

While the distro doesn't ship with any proprietary codecs, you can watch YouTube videos as it provides HTML5 support as well as Gnash, which is the free alternative to Adobe Flash. Trisquel includes all the usual desktop productivity apps, such as *LibreOffice*, *Evolution*, *Gwibber*, *Pidgin* and more. These are complemented by an impressive software repository.

Genus Red Hat

Millinery on a massive scale.



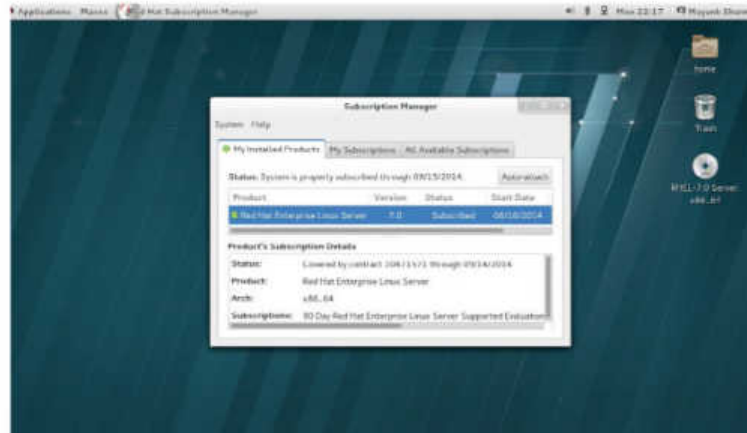
Another distribution that has played a crucial role in shaping Linux's DNA is Red Hat Linux, which was created in 1994 by Marc Ewing. Bob Young and his ACC Corporation bought Ewing's business and created Red Hat Software. The company went public in 1999 and achieved the eighth-biggest first-day gain in the history of Wall Street. It rode on the success of Red Hat Linux to become the first open source billion dollar company.

Over the years, some of the biggest and brightest Linux developers have worked with Red Hat. Soon after it went public, it acquired Michael Tiemann's Cygnus Solutions which had authored the GNU C++ Compiler and worked on the GNU C Compiler and the GNU Debugger.

One of Red Hat's most influential pieces of technology is its RPM packaging format. The file format is now the baseline package format of the Linux Standard Base (LSB), which aims to standardise the software system structure, including the filesystem hierarchy used in the Linux operating system. The LSB is a joint project by several Linux distros managed by the Linux Foundation. Red Hat was also one of the first Linux distros to support Executable and Linkable Format (ELF) instead of the older a.out format. ELF is the standard file format for executables, shared libraries and other files. Red Hat was also the first distro to attempt to unify the look of its Gnome and KDE desktop with the Bluecurve theme – which caused tension with the KDE developers. The distro has won laurels for its easily navigable graphical Anaconda installer.

Life after death

Initially, the Red Hat distro was offered as a free download and the company sustained itself by selling support packages. In 2003, however, Red Hat discontinued the Red Hat Linux distro and it now focuses solely on the Red Hat Enterprise Linux (RHEL) distro for enterprise environments. RHEL supports popular server architectures including x86, x86-64, Itanium, PowerPC and IBM System z. The lifecycle of newer RHEL releases spans 13 years, during which time the users get technical support, software updates, security updates and drivers for new hardware. Red Hat also has a very popular training and certification program called RHCP that's centred around RHEL.



When Red Hat Linux was discontinued, the company handed over development of the free distro to the community. The new project was called Fedora (see p36). The company steers the direction of the Fedora project and does so in order to use Fedora to incubate technologies that will eventually show up in RHEL.

Since the GPL prohibits it from restricting redistribution of RHEL, the company uses strict trademark rules to govern the redistribution. This has led to popular third-party derivatives that are built and redistributed after stripping away non-free components like Red Hat's trademarks. Distro such as CentOS, Scientific Linux and Oracle Linux offer 100% binary compatibility with RHEL.

Red Hat has served as the starting point for several other distros, such as Mandriva Linux.

“Some of the biggest and brightest Linux developers have worked with Red Hat.”

Red Hat has pioneered the professional open source business model, successfully mixing open source code and community development together with professional quality assurance, and a subscription-based support structure. The company also has employees working full-time on free and open source projects, such as Radeon, Nouveau and CentOS.



**BEST
OF
BREED**

CentOS

The CentOS distro has been the premier community-supported enterprise distro based on Red Hat Enterprise Linux (RHEL). The distro is built using the open source SRPMS from the RHEL distro. CentOS is one of the most popular server distros, suitable for all kinds of use cases, from web servers to enterprise desktops, and has been able to pitch itself as an

ideal choice for anyone who wants to put together their own server but can't afford the RHEL subscription fees.

CentOS ships with RHEL's *Anaconda* installer and can run unattended installations across multiple machines thanks to *Kickstarter*. The installer provides various installation targets such as a web server, database server etc.

In January 2014, Red Hat announced that it would start to sponsor a bunch of core CentOS developers to work on the distro full-time. However, the developers and Red Hat have both insisted that the project remain independent of RHEL. The sponsorship ensures that all updates will be provided within 24 to 48 hours of upstream releases in RHEL.

Genus Fedora

You've been hit by a smooth distro.



» **F**edora has been around, in one form or another, since the early 1990s. The distro had its first release in 1995 and the early releases were named Red Hat Commercial Linux. During these early years, the distro was developed exclusively by Red Hat and the community was limited to contributing bug reports and contributing packages included in the distro. This changed in 2003 when the company shuttered Red Hat Linux in support of the Fedora Project and opened it up to contributions from the community.

The aim of Fedora is to provide the latest packages while maintaining a completely free software system. The distro was initially called Fedora Core and was named after one of the two main software repositories – Core and Extras. The Fedora Core repo contained all the basic packages required by the distro as well as other packages distributed with the installation discs, and was maintained exclusively by Red Hat developers. The Fedora Extras repo was introduced with Fedora Core 3. It contained packages maintained by the community and was not distributed with the installation discs. This arrangement continued until version 7 in 2007 when the two repos were merged and the distro was renamed as Fedora.

Fedora's objective is to create a free software distribution with the help of the community. The development of the project is overseen and coordinated by the Fedora Project. It's

made up of four Red Hat appointed members and five community elected members. The chairman of the board is appointed by Red Hat. Fedora strives to maintain a roughly six-month release cycle, with two releases each year. Every release is supported until the launch of the next two releases. The cycles are deliberately kept short so that developers can focus on innovation and introducing the latest technologies into the distro.

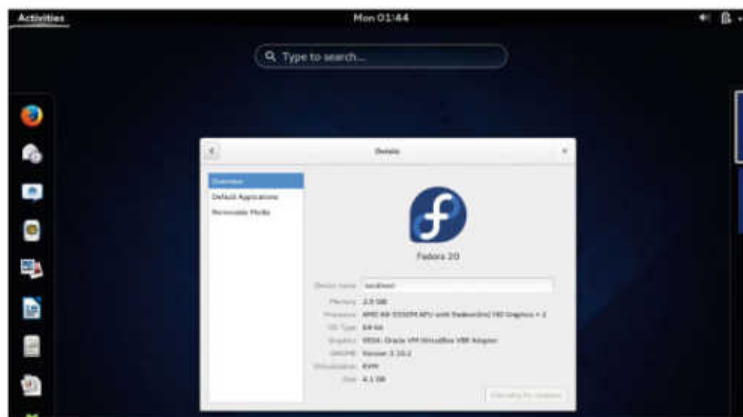
Feather in the cap

One way the community contributes is by hosting third-party repositories. In addition to its official software repos, there are several popular third-party software repos that usually contain software not included in the official repos – either because of the current laws of the country (such as multimedia codecs) or because the software doesn't meet Fedora's definition of free software. The Fedora project also produces the Extra Packages for Enterprise Linux (EPEL) repo, which contains packages for RHEL that are created by the community instead of Red Hat.

Apart from the main Fedora release, the project also ships various spins which are special-purpose distros aimed at specific interests, such as gaming, security, design, scientific computing etc. These are similar to Debian's Pure Blends. These and others are maintained by various Special Interest Groups (SIGs). The OLPC also runs a Fedora-based operating system. Fedora supports the x86 and ARM architectures and has also added support for PowerPC and IBM s390, starting with Fedora 20. Pidora is a Fedora Remix distro optimised for the Raspberry Pi.

Fedora's biggest contribution to the Linux ecosystem is its command line package manager, *YUM* (Yellowdog Updater, Modified), which is based on *RPM* (Red Hat Package Manager). *YUM* enables automatic updates and dependency resolution, and works with the software repositories to manage the installation, upgrading and removal of packages. Since the release of Fedora 18, users have had the option to use the *dnf* tool which is a fork of *YUM*. The *dnf* tool will likely become the default package manager in Fedora 22, because it has better dependency resolution and is less memory intensive than other managers.

» **Fedora was one of the first distros to embrace the Security Enhanced Linux (SELinux) kernel module.**



**BEST
OF
BREED**

Korora

The Korora distribution started out as a way to ease the installation process of the Gentoo distro, but switched to using the Fedora distro as the base in 2010. The main aim of the distro is to make sure it works right out-of-the box for users.

Korora ships a live DVD, which includes a huge selection of apps that make it suitable for a large number of

users and the distro offers five desktop choices – Gnome, KDE, Cinnamon, Xfce and Mate.

While Fedora only ships with open source software, Korora also includes some proprietary software, such as Adobe Flash, which are essential for catering to a wide user base. Korora also allow other software to be easily installed, such as *Google Chrome* and

the proprietary graphics driver for Nvidia cards.

The distro has also eased a gripe for some Fedora users: graphical package management. Korora includes both *Apper* and *Yum Extender*, which are two of the most popular front-ends for *YUM*. Since it's based on Fedora, a new version of Korora is usually a few weeks behind a Fedora release.

Genus Mandrake

A distro which has a lot to scream about.



Until the release of Mandrake, Linux was generally thought of as a geek's OS. Mandrake was the first distribution that focused on the convenience of the user. The goal was to provide a distro that could be operated by regular computer users. It had features such as the ability to auto-mount CDs without messing around with configuration files, which brought greater convenience to the Linux desktop.

The Mandrake project has perhaps the most convoluted existence for a free software project. Over the years, the project has undergone various name changes, mergers and forks. However, it has spawned many distros and there are several major ones that are still active and can trace their lineage to Mandrake.

The distro has developed a bunch of custom tools – collectively known as drakxtools – to aid its users, who are called Drakes or Draks. One of the most distinguishing components created by the project is its *Mandrake Control Center* (MCC), which is now a centrepiece of all the derivatives. The MCC provides a single interface to access many different configuration tools. Using the *Control Center* in text mode is very useful in case of display problems or other serious issues, such as when the graphical server refuses to start. It's also interesting to note that all modules can be run as autonomous apps, without necessarily having to go through the MCC.

On life support

Mandrake Linux was first released in July 1998. It was based on Red Hat Linux 5.1 and featured the inaugural KDE desktop release. After the positive response, lead developer Gaël Duval, along with a bunch of others, created the company MandrakeSoft and in 2001 the company decided to go public. It faced its first major cash issue in late 2002 and asked its users to bail it out by subscribing to a paid service offering extra benefits, such as early access to releases and special editions. That wasn't enough and the company filed for bankruptcy protection in 2003. However, later that year MandrakeSoft announced its first quarterly profit and, in March 2004, a French court approved its plan to emerge from bankruptcy and return to normal operations.



The company also had to rename its product to Mandrakelinux, after losing a legal battle with the American Hearst Corporation over the Mandrake name and changed its name to Mandriva S.A. after acquiring the Brazilian company Conectiva in 2005. The distro became Mandriva Linux, but in 2006 Mandriva laid off several employees, including co-founder Duval. Amidst all the booing, the company continued putting out releases and created a niche for itself in the BRIC region (Brazil, Russia, India and China) as well as France and Italy.

Despite all their efforts, the company struggled to keep its balance sheet in the black. In 2010, Mandriva abandoned development of the community Linux distro to avoid

» Rosa Linux is a Mageia fork that features some dramatic user interface mods such as this Simple Welcome app launcher.

“One of the most distinguishing components created by the project is its Control Center.”

bankruptcy. Immediately afterwards, former Mandriva employees announced Mageia, which has gone on to be one of the most popular Mandrake-derivatives. Mandriva S.A. transferred development to the community-driven Open Mandriva Association. The association's second release called OpenMandriva 2014.0 got a positive review from Duval. »

**BEST
OF
BREED**

Salix Mageia

Mageia is one of the best-assembled community distros and does a wonderful job of carrying forward the Mandrake legacy. It has an expansive support infrastructure and very good documentation. The distro follows a nine-month release cycle and each is supported for 18 months. Mageia has installable live media as well as install-only DVD images.

Mageia boasts intuitive custom tools for managing various aspects of the distro. One of the best tools is the *Mageia Control Center*, which has modules for managing software, hardware peripherals and system services. Advanced users can employ it to share internet, set up a VPN and configure network shares. The distro uses the URPMI package management

system and ships with three official repos. The Core repo contains open source packages, the Non-free hosts proprietary apps and drivers, and the Tainted repo includes patent-encumbered apps. The distro ships with several desktop environments, and the developers have made sure the user experience is consistent across all of them.

Genus SUSE

Nuga, nuga, nuga, gnu, nui.*

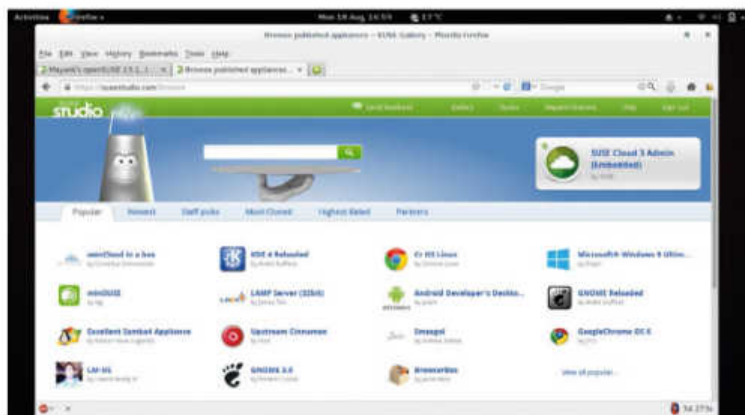


» In 1992 Roland Dyroff, Burchard Steinbild, Hubert Mantel and Thomas Fehr founded Software und System Entwicklung (Software and Systems Development). The company started as a service provider but the founders decided to have a distro of their own to cater to the enterprise user. The distro was named SUSE, based on the acronym of their company. The distro was a stock Slackware release translated in German and developed in close collaboration with Slackware's Patrick Volkerding.

For building its very own distribution of Linux, SUSE used the now defunct Jurix distribution. Jurix was created by Florian La Roche, who subsequently joined the SUSE team and began to develop YaST, which is the distro's unique installer and configuration tool. The first SUSE distro that included YaST was released in May 1996 (YaST was rewritten in 1999, and was included for the first time in SUSE Linux 6.3 as an installer only).

Over time, SUSE Linux has incorporated many aspects of Red Hat Linux, such as its well-respected RPM Package Manager. In 1996, the first distribution under the name SUSE Linux was published as SUSE Linux 4.2. The confusing jump forward in version numbers was an intentional reference and homage to the answer to life, the universe and everything, as featured in Douglas Adams' *The Hitchhiker's Guide to the Galaxy*. YaST's first version number, 0.42, was inspired by the same admiration for the author.

» The SUSE Studio web service enables you to easily put together a customised OpenSUSE-based distro.



SUSE's focus has always been on bringing open source to enterprise users. It introduced the SUSE Linux Enterprise Server in 2001, and changed the company name to SUSE Linux. After software and services company Novell acquired SUSE Linux in January 2004, the SUSE Linux Professional product was released as a 100% open source project and the OpenSUSE Project was launched – much like Red Hat did with Fedora. The software was always open source and now so was the process which enabled developers and users to test and evolve it.

Enterprising

The initial stable release from the OpenSUSE Project was SUSE Linux 10.0. It included both open source and proprietary applications, as well as retail boxed-set editions. This was also the first release which treated the Gnome desktop environment on a par with SUSE's default KDE desktop. As of version 10.2, the SUSE Linux distribution was officially rechristened as OpenSUSE.

In November 2006, Novell signed an agreement with Microsoft covering improvement of SUSE's inter-operability with Windows, cross-promotion and marketing of both products, and patent cross-licensing. This agreement is considered controversial by some of the FOSS community.

Novell was later acquired by The Attachmate Group in 2011, which then divided Novell and SUSE into two separate subsidiary companies. SUSE offers products and services around SUSE Linux Enterprise – a commercial offering that is based on OpenSUSE Linux.

SUSE develops multiple products for its enterprise business line. These products target corporate environments and have a longer lifecycle (seven years, extendable to 10), a longer development cycle (two to three years), technical support and certification by independent hardware and software vendors. SUSE Linux Enterprise products are only available for sale. There's also the SUSE Linux Enterprise Desktop (SLED) which is a desktop-oriented operating system designed for corporate environments. In contrast, OpenSUSE does not have separate distributions for servers, desktops and tablets, instead using various installation patterns for different types of installation.

*<http://bit.ly/ChameleonSong>



OpenSUSE

OpenSUSE is one of the best RPM-based distros. It comes in several editions for 32-bit and 64-bit architectures and also has ports for ARM v6, ARM v7, and the 64-bit ARM v8. Once known for its KDE desktop, OpenSUSE now looks good across all the major desktops. Besides KDE and Gnome, the distro also features Mate, Xfce, Enlightenment, and LXDE. You

can download the distro either as a smaller live installable image or a install-only DVD image.

One of OpenSUSE's hallmarks is the distros YaST, which is a setup and configuration utility that enables you to tweak many different aspects of the system. Another popular tool is Snapper, which enables you to revert to a previously created system snapshot.

The distro serves as the base for the SUSE Linux Enterprise products – much as Fedora does for RHEL – and is suitable for all types of users regardless of their skill set. The distro's installer is versatile and offers several customisation options. It can be navigated by new users and includes options to plug the installed system into a corporate directory server.

Genus Slackware



The tortoise distro that's outlasted many hares.

Slackware has the honour of being the oldest distro that's still actively maintained. It was created by Patrick Volkerding and had its first beta release in 1993. The project aims to create the most Unix-like Linux distribution. Slackware was originally derived from Softlanding Linux System (SLS), which was the first distro to provide TCP/IP and X Windows System in addition to the Linux kernel and basic utilities. SLS, however, was very buggy and the growing frustration of SLS users prompted Volkerding to release an SLS-like distro in July 1993.

Back then, in addition to being hosted on an anonymous FTP server at the Minnesota State University Moorhead, the distro was offered as 24 3.5-inch floppy disks. By the time version 2.1 was released in October 1994, the distro had swelled to 73 disks and Version 3 was released on CD-ROM.

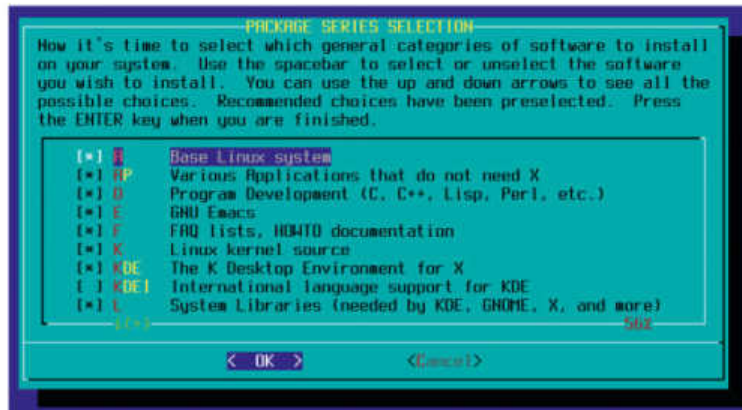
The USP of the distro is that it makes very few changes to upstream packages. Unlike other distros that aim for a particular userbase or a wide variety of users, Slackware doesn't preclude user decisions and doesn't anticipate use cases. The user, therefore, has far greater control on the installed system with Slackware than most other distros.

Cut some slack

Unlike other distros, Slackware doesn't provide a graphical installation. It continues to use plain text files and only a small set of shell scripts for configuration and administration. The distro also doesn't provide an advanced graphical package management tool, relying instead on command line tools such as *pkgtool*, *installpkg*, *upgradepkg*, and *removepkg*. However, these native tools can't resolve dependency issues.

Slackware packages are just plain compressed TAR archives. The package contains the files that form part of the software being installed, as well as additional metadata files for the benefit of the Slackware package manager. As of Slackware 12.2, *slackpkg* has become the official tool for installing or upgrading packages automatically through a network or over the internet, complementing the traditional package tools suite that only operates locally. *Slackpkg* also doesn't resolve dependencies between packages.

Traditionally, Slackware only offered a 32-bit release, and users had to rely on unofficial ports, such as *slamd64* for



64-bit releases. Since Slackware 13, a 64-bit variant is also available and officially supported. In 2002, Stuart Winter started the ARMedslack project, a port of Slackware for ARM. In 2009, Volkerding knighted ARMedslack as an official port of Slackware. With the release of Slackware 14.0, the project has been completely renamed to Slackware ARM.

It might sound surprising, but Slackware is a popular base for many distros. The derivatives include expansive desktop projects, live distros, security distros etc.

The Slackware project is also missing some of the common developer-friendly tools. For example, there's no official bug tracking system. Also, there is no official mechanism to become a contributor for Slackware. The final

» In addition to Slackware-stable, the project also provides a testing-current branch for more bleeding-edge software.

“Unlike other distros, Slackware doesn't preclude user decisions and doesn't anticipate use cases”

decision on what goes into the distribution is made by Volkerding – Slackware's 'Benevolent Dictator For Life'.

In another departure from the norm, Slackware doesn't follow a fixed release schedule. The objective is to release a very stable system and so the project follows a release-when-ready philosophy, but still aims for one major release a year. »

**BEST
OF
BREED**

Salix OS

Salix OS is one of the best Slackware-based distros: it's light, nimble and backwards compatible with Slackware. One of its salient features is that it minimises bloat by having only one application per task. The distro supports both 32-bit and 64-bit architectures and is available in five variants for the KDE, Mate, Xfce, Openbox, and Ratpoison desktops.

Salix offers three modes of installation – Full, Basic and Core. The Full option installs everything on the installation image; Basic provides a barebones system with just the graphical desktop and a few essential apps and the *Slapt* package manager; the Core option will only install a console-based system and is designed for users to custom-build their install.

The Full distro includes all the apps you'd expect on a desktop distro, and is often touted as Slackware with a graphical package manager. Its package manager, *Gslapt*, resembles the *Synaptic* package manager and also provides all the same functionality. Multimedia codecs aren't supplied out of the box, but that can be fixed with the distro's custom *Codecs Installer*.

The marsupials

Evolutionary masterpieces and mavericks.

» Gentoo Linux

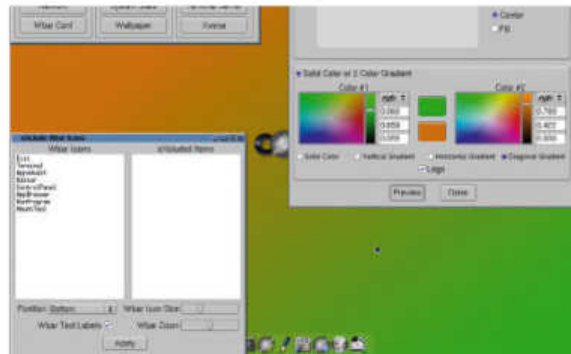
The goal of the Gentoo project was to create a distro without pre-compiled binaries that was tuned to the hardware on which it was installed. Unlike a binary software distribution, the source code is compiled locally according to the user's preferences and is often optimised. It was initially called Enoch but Gentoo 1.0 was released in 2002.

Gentoo has the distinction of being one of the most configurable distros and appeals to Linux users who want full control of the software that's installed and running on their computer. Gentoo users get to create their system from the ground up: the distro encourages the user to build a Linux kernel tailored to their particular hardware. It allows very fine control of which services are installed and running. Memory usage can be reduced, compared to other distributions, by omitting unnecessary kernel features and services.

This distro is a rolling release and one of its notable features is its package management system called *Portage*. If you've never used it before, there's a steep learning curve to using Gentoo. Derivatives such as Funtoo can be a good starting point if you're not ready to dive straight in.

Arch Linux

Judd Vinet wanted to create a distro that was inspired by the simplicity of Crux, Slackware and BSD and thus created Arch Linux in 2002. Arch aims to provide a lightweight foundation on which the user can build according to their needs. In Vinet's words: "Arch is what you make it". A bit like life really.



» **Tiny Core Linux is an ickle distro at 12MB. Ah, sweet.**



» **Arch Linux is ludicrously customisable, offering all the latest packages and no cruft.**

The most impressive feature of the Arch distro is the *Pacman* package management tool. Arch is a rolling release that can be brought up to date with a single command. Installing Arch Linux is an involved process and although it is well-documented, it's still better suited for experienced Linux campaigners. However, Manjaro Linux is an Arch derivative and is more user-friendly and has a graphical installer.

Tiny Core Linux

If you can't invest time in creating an Arch or Gentoo installation, check out Tiny Core Linux. The distro installs the bare minimum software you need to boot into a very minimal X desktop. From this point on, you've complete control and can install apps from online repos or compile them manually.

The distro is a mere 12MB and bundles only a terminal, a text editor and an app launcher on top of the lightweight *FLWM* window manager. It has a control panel to manage bootup services and configure the launcher, but everything else needs to be pulled in from its manager, including the installer if you want Tiny Core on your hard disk. The distro also has a CorePlus variant, which has additional drivers for wireless cards, a remastering tool and internationalisation support.

Puppy Linux

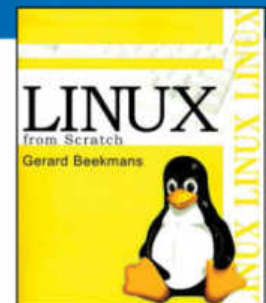
One of our all-time favourites, Puppy Linux had its initial release in 2003 and the first stable one in 2005. The distro is built from the ground up and its initial goal was to support

Linux From Scratch

Rather than being a distribution itself, Linux From Scratch – popularly called LFS – is a freely available set of instructions to create your own custom distro from the ground up, entirely from source. The project was started in 1999 when its author, Gerard Beekmans, wanted to learn how a Linux distro works behind the scenes. While building his system from scratch, Beekmans wrote down the steps and released it as a HOWTO (pictured, right)

thinking that there would probably be other people who would be interested.

LFS has grown quite a bit from its humble start, transforming from a single HOWTO to a multi-volume book. It has also spawned various sub-projects over time, such as BLFS or Beyond LFS which fleshes out the basic LFS system, and ALFS or Automated LFS, which is designed to help automate the process of creating an LFS system.





► **Puppy Linux** has become a handy distro for recovering data from PCs and removing malware from Windows.

older hardware that had been rendered useless due to lack of support in other distros.

The real power of the distro lies in its plethora of custom apps. There are custom apps to block website ads and add internet telephony, a podcast grabber, a secure downloader, an audio player, a DVD burning app and more. First-time users might be intimidated by Puppy's installer as it has no automatic partitioner, and fires up *Gparted* for you to format the disk. But each step in the installer is well-documented within the installer itself.

Packages for Puppy Linux are called pets, and have a .pet extension. You can install packages using its custom *Puppy Package Manager* tool, and you can configure it to download packages from other Puppy repos. The distro includes tools which can be used to easily churn out variants. Puppy Linux variants are called puplets. Popular puplets are WaryPuppy for supporting older hardware, RacyPuppy for newer hardware, the Slackware-based SlackoPuppy, and PrecisePuppy which is based on the Ubuntu LTS release.

SliTaz GNU/Linux

SliTaz stands for Simple Light Incredible Temporary Autonomous Zone and had its first stable release in 2008. The distro is built with home-brewed tools known as cookutils and uses *BusyBox* for many of its core functions. The distro includes a mixture of the LXDE and *OpenBox* window manager and is designed to perform on hardware with only 192MB of RAM. The distro weighs under 30MB and uses a mere 80MB of hard disk space.

The distro also has a bunch of custom tools such as the *Tazpkg* package manager and *SliTazPanel* for administering all aspects of the distro. SliTaz repos include over 3,000 packages for every popular open source app and it's a common option for powering low-powered machines.

PCLinuxOS

PCLinuxOS began life as a repository of RPM packages for the Mandrake distro in 2000 and became a distro in late 2003 as a branch of Mandrake Linux 9.2.

Although it retains a similar look and feel to Mandriva Linux, and its configuration tool and installer give away its Mandriva lineage, PCLinuxOS has diverged significantly.

The distro has replaced Mandrake's *URPMI* package management system, opting instead for APT-RPM. This is based on Debian's *APT* but uses RPM packages, together with the *Synaptic* package manager. PCLinuxOS is a KDE distro, but also has community spins around the LXDE and Mate desktops.

TOP DESKTOP DISTRO

We're going to tread on that hallowed patch of earth where angels fear to tread (and not because a bushy-bearded Russell Crowe is eyeballing them) and attempt to pick an overall distro winner. This means we had to pick a criteria that allowed for ease of use alongside the ability to build in complexity for specific use cases and, we admit, the result is purely subjective. Don't agree with us? Why not email your top picks for each genus to Linux Format magazine at lxformat@futurenet.com.

1

Mageia



The community supported distro has everything you want from a modern Linux distribution – an active and vibrant user and developer community, a well-defined support structure, support for multiple desktops and install mechanisms.

2

OpenSUSE



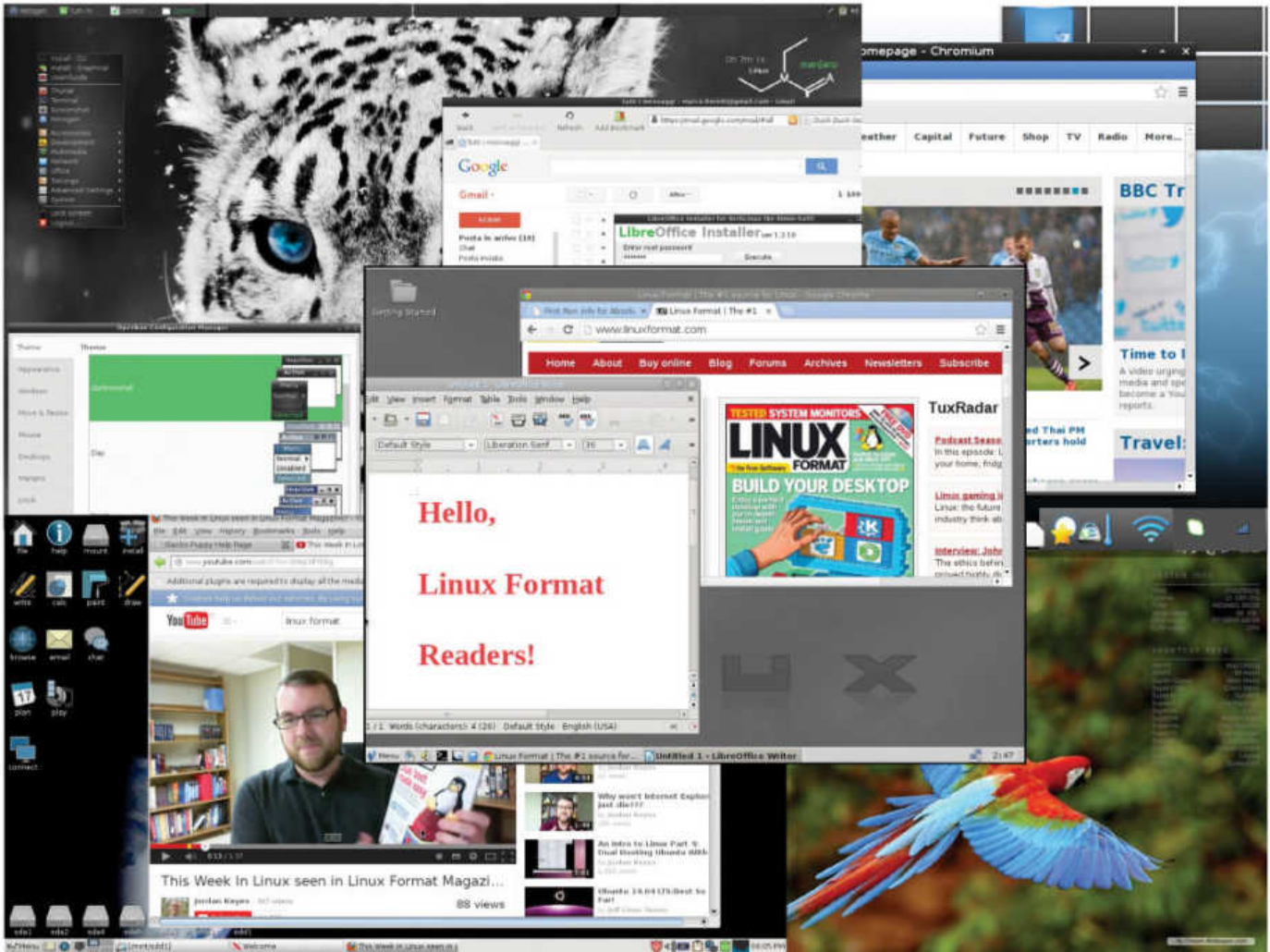
Coming in at second place, the OpenSUSE distribution loses out because of recent activities of its corporate parent. Also, the distro still focuses on introducing changes that make it fit more snugly on the corporate desktop, rather than home user.

3

Korora



This is your best bet if you want a RPM-based distribution that works out of the box. However, Korora is still essentially a one-man show and inherits some of the less flattering features of its parent distro, Fedora.



Low-resource distros

Do low-resource Linux desktops for the average user still exist? We test five less well known distributions that are fast and not so furious to use.

Our selection

- » Absolute Linux
- » Crunchbang2
- » Elive
- » Manjaro
- » Puppy

Computer hardware has made huge progress in the last 20 years. So have Linux desktops, but in a different way. Modern computers – in whatever format, from smartphones to laptops – are orders of magnitude more powerful than their predecessors of the 90s. But Linux desktops have not become tens of times faster. So what options do you have if you want a faster Linux, particularly on older hardware?

Many Linux developers produce so-called 'low-resource' distros, which require much less RAM and processor cycles than Ubuntu and similar state of the art products. Unfortunately, this too often creates graphical desktops that

are useless for most people. Damn Small Linux (DSL), for example, is great for software engineering, but how many mainstream things can it do? Can you really use DSL to edit an Excel spreadsheet, browse YouTube clips, use Ajax-based cloud services such as Gmail, or easily upgrade software?

This is what led us to this roundup. We did not want the smallest possible Linux desktop, or the one that will boot with the least RAM. We looked, instead, for distros in which ordinary users, without advanced Linux skills, can work with office documents, check their email and use popular web services – even on older computers – with the least possible effort.

How we tested...

We did not consider Lubuntu and other 'light' Ubuntu spin-offs, simply because they are the ones that most Linux novices who need a low-resource distro are more likely to know already. After a lot of investigation, we chose one distro with a truly unique architecture (Puppy) and four that are lightweight derivatives of more popular systems, namely Slackware, Arch Linux and Debian. We then ran them live on a low-end laptop and/or installed them in virtual machines with different RAM sizes, but never more than 2GB. In all cases, we tested the systems with ordinary day-to-day desktop activities, from browsing and using office suites to displaying photographs. We also tried some of the usual system administration tasks, from installing software to adding users and changing the general system configuration.

Desktop completeness

What can you actually do with this distro?

The main reasons why many people interested in Linux never try a 'low-resource' distro are probably lack of need, support and apps. For somebody who just got a high-end computer, it would be easy to not care. Newcomers to Linux also often stay clear of 'niche' distros because, understandably, they want to remain where there are more tutorials and fellow users. We discuss that issue overleaf. Here, let's test that 'lack of apps' complaint.

The bottom line is that, for the distros here, there is no 'lack of apps' and your real limit is going to be the hardware. No choice of software will let you edit high-definition video, or process spreadsheets with thousands of rows, in (say) 128MB of RAM.

As far as traditional SOHO tasks go, you'll find apps to do almost everything you want with any of these desktops, sometimes the same tools available on more famous Linux desktops. If an app is missing, no problem – you should still be able to install it from predefined archives with a few clicks or keystrokes.

The default dock in Elive is pretty impressive; the tested version is still at *LibreOffice 3*, but it's accompanied by *Clementine*, *Skype*, *Scribus*, *Chromium*, *Blender*, *Inkscape*, *RawTherapee*, *Gimp* and *VLC*. And that's just what you find in plain view in the dock.

Puppy is also impressive, but in a very different way. Some versions come with *SeaMonkey* as browser, email client and RSS reader. In Slacko Puppy we found *Firefox* and the *Sylpheed* email client.

Besides that, the Slacko menu is built, it seems, in the good old Unix tradition, with so many – often little-known – small tools that we lost count, often with a vintage look and interface, each specialising in one activity. Combining them, you can do pretty much whatever you want, if you take the time to learn. The default word processor is *AbiWord*, but adding *OpenOffice 3.1* is easy.

Absolute, Crunchbang and Manjaro are broadly in the middle ground between these two extremes, each in its own way. In Absolute you'll find several



▶ The Elive system menu is simple, but between that and the dock, there really is everything you need.

specialised development applications (*HtmlPage*, *Qt Designer* and a hex editor) and can get *Libre Office 4.1.4*.

The Internet apps menu includes *Filezilla* and a YouTube viewer.

Crunchbang comes with the *IceWeasel* browser, which works fine, even if it is a little stuttery at times. The Multimedia choices are limited to volume control, *VLC* and *Xfburn* for DVD mastering. Manjaro, which contains *Libre Office 4.1.5*, is in a similar situation – *Alsa* mixer and little else. Apart from this, it is the only distro here that couldn't play YouTube videos out of the box.

Verdict

Absolute
★★★★★
Crunchbang
★★★★★
Elive
★★★★★
Manjaro
★★★★★
Puppy
★★★★★

» Elive has the fullest set of apps, but others can pretty much do it all too.

Installation

Is it easy to get started with these systems?

In general, the installation of these five distros, or their usage as live systems, should not present any particular problems, unless you put them on very new, more or less closed hardware, or on complex combinations of hard drives.

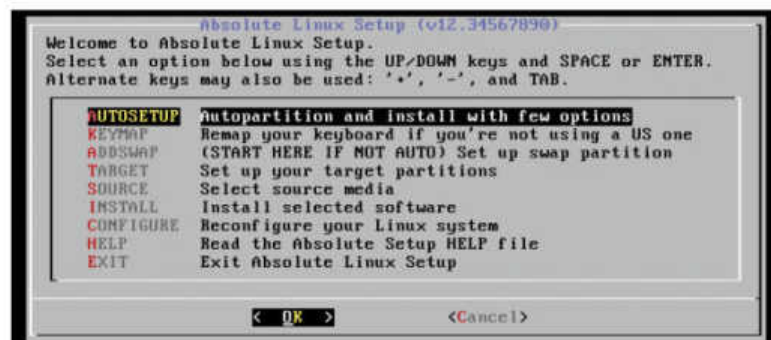
The installer of Absolute Linux, for example, is text-only, and uses the LILO

bootloader. If you didn't know Linux existed in 1994, try Absolute to see what installing it looked like. Don't worry, though – the look may be ugly, it's all keyboard and no mouse, and it takes more time than you might like to read through some panels, but there is little real difficulty involved. The hardware detection works fine and you can safely

accept the default settings and answers to each question that you don't understand immediately.

The most friendly installers (or boot managers, for live versions) are the graphical ones of Manjaro and, above all, Elive. The former, called Thus, is maybe the one that looks most like Ubuntu. It may also encrypt partitions. The latter has a very polished language and keyboard selector, plus interesting options (see 'Choices' on page 93).

The Slacko Puppy boot manager doesn't put any special effort into eye candy, but works without problems. Besides, most of the actual system configuration is delegated, after login, to a friendly set-up wizard. That is where you must set country encoding, time, keyboard, network settings, graphics stuff and other parameters. And it even barks in approval when you're done.



▶ Absolute Linux's installer makes you ask what year it is but isn't hard to use.

Verdict

Absolute
★★★★★
Crunchbang
★★★★★
Elive
★★★★★
Manjaro
★★★★★
Puppy
★★★★★

» Absolute's installer is basic but Elive's is the one that'll make novices happy.

User interface

Is it customisable, attractive, but not too resource-hungry?

The interface of any computing device, from smartphone to traditional desktop, is a balance of conflicting requirements: minimal consumption of RAM and CPU cycles, which in our case is a must, usability by non-experts, room for customisation, and finally

good looks (whatever 'good' might mean for each user).

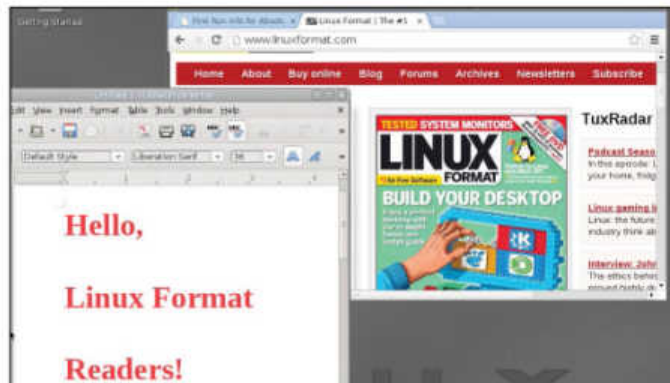
After testing, we were pretty happy with our choice of distros. The reason is that each of them gives a different but equally valid answer to the requirements above. Some people will

prefer the easy but infinite visual configuration possibilities of Elive, others the rich but relatively rigid structure of Absolute or Slacko. Others still will love the no-frills approach of Manjaro and Crunchbang. But there should be something for everybody.

Absolute ★★★★★

Absolute uses the IceWm window manager to give you a fast and discreet desktop. A right-click on the root window, which by default would launch the tool to change wallpaper and screen resolution, can be set to open the system menu.

The file manager is *SpaceFM*, not state of the art by any means, but adequate for all basic tasks. Of all the distros in this roundup, Absolute is perhaps the one that looks more like a basic Linux of 15 years ago. Just for this reason, however, it may be perfect for many users, especially when you consider that it's just a first sight impression – you do get a modern browser (*Chrome*) and office suite (*Libre Office*), and often that's all that counts, isn't it?



Crunchbang ★★★★★

The first time you start it, Crunchbang is all business and no frills. All you see is a visually dull but well configured OpenBox window manager with two workspaces, a system monitor and the list of shortcut keys. However, changing the wallpaper – if a solid grey screen really bothers you – is a quick and easy job.

One right-click on the root window opens a menu with apps for all desktop tasks. That menu, plus the *Terminator* terminal emulator and the *Thunar* file manager, let you work with the mouse or keyboard very efficiently. Crunchbang is also the distro most interested, so to speak, in working in the cloud with common tools; the system menu includes entries to start Google Docs and install the Dropbox client.

Documentation

Is it easy to learn how these distros work?

If you need Linux just to run a few apps on an older computer, the mere number of available pages of documentation may matter much less than two other things. First, check if those apps, rather than the distro, have good documentation. Second, look for distros that hold your hand well in the first two to three hours of usage, when you really need to get familiar with them and their community.

Crunchbang's default wallpaper is what all distros for newcomers should

have: a cheatsheet with all the main keyboard bindings. Most Debian material will apply to Elive, too, so if anything, there may actually be too much documentation.

Manjaro and Absolute are the best here. At first login, Manjaro welcomes you with a window full of links to documentation, online support and descriptions of the available desktops.

The 'Getting Started' icon in the Absolute desktop loads a 14-point, task-oriented guide, with sections such as

'Multimedia files', 'Wireless setup', adding software and Nvidia drivers. The Documentation menu links to the manuals of the distro and its main packages. The same happens in Puppy.

Speaking of Puppy, note that many of its online guides and tutorials may only be valid for a specific version (read 'Choices' opposite to understand why). Therefore, before trying any Puppy, ask at www.murga-linux.com/puppy what are the best and most up-to-date resources for that version.

Verdict

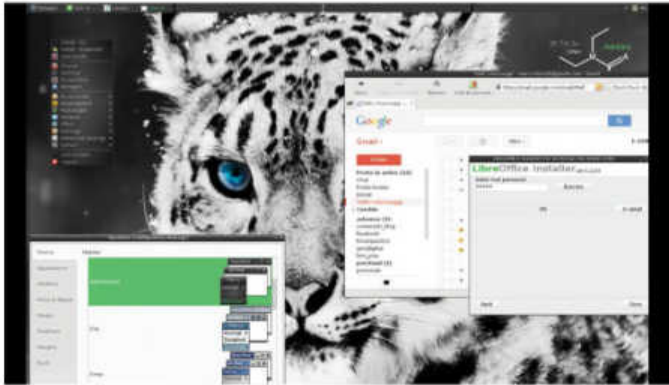
Absolute ★★★★★
Crunchbang ★★★★★
Elive ★★★★★
Manjaro ★★★★★
Puppy ★★★★★

» All have OK documentation. In Manjaro and Absolute it's easier to reach.

Elive ★★★★★

Elive is a great showcase for the Enlightenment desktop. In the 90s, when it appeared, people (including us) would have rolled on the floor laughing if told that one day Enlightenment would be a low-resource choice. Today, we can testify that Elive gives the best compromise between visual effects and performance on limited hardware.

The system monitor, with its gauges for battery charge, CPU temperature and power policy, is attractive as well as useful, and there are enough apps, including the *Unison* file synchronisation tool. If you have the patience to configure it, everything will look just as you want, from the font of window titles to sticky notes. Before that, though, turn off the fading and zooming effects if you're really short of RAM.



Manjaro ★★★★★

If distracted users confused Manjaro with Crunchbang, they would be justified – both systems are Arch Linux spin-offs dressed in OpenBox. The main difference, besides a smaller selection of apps, is in the default browser. Manjaro uses *Midori*, which is enough to run Gmail and similar web services based on Ajax, but it can't play YouTube videos.

On the other hand, Manjaro includes tools that you may never have heard of before, from the *Parcellite* clipboard manager to *Avahi* browsers for SSH and VNC servers. Manjaro also makes it easier than Crunchbang to fiddle with OpenBox, especially if you already know how to handle it. Besides the Tint2 configurator, the system menu contains all the possible tools to configure the workspaces and dynamic menus of this powerful window manager.

Puppy ★★★★★

Slacko 5.7, the member of the Puppy pack we're looking at in this context, is at the time of writing the "recommended first stop for all newcomers" to the Puppy family. Its user interface is the result of a careful mix of the JVM window manager, the *ROX Filer* and many big and small apps, from household names such as *Firefox* to really obscure utilities. The System sub-menu even has a floppy disk formatting utility, together with the *GParted* partition manager.

We don't understand the rationale behind separate sub-menus for 'Document', 'Business' and 'Personal' applications, but there they are. This said, *JVM* and *ROX* work well together, and *ROX* is one of those apps that every Linux user should try at least once – fast, simple and cute. Lean and mean doesn't mean ugly or lacking in features.



Choices

What if the version tested here is almost what you want?

Are these distros a 'take it or leave it' deal? It depends on what you'd like to change. If all you want is alternative window managers, no problem. Absolute is declared 'package compatible' with Slackware releases with the same number, while Crunchbang and Manjaro have access to the Arch repositories; you should find enough packaged, easily installable alternatives to make you happy. Ditto for Elive, which may reuse stuff from Debian and

also has an Experimental Mobile mode for touchscreen phones, tablets and similar devices.

Manjaro has three official variants, all with Tulliana and KFaenza icons: two for Xfce (full or stripped-down) and one with KDE 4.12 (with KDE Connect <http://community.kde.org/KDEConnect>), deliberately configured to "give all Windows users a new home".

Want more? Try the Manjaro community editions for other desktop environments (Mate, Cinnamon, LXDE

and more), the netbook spin for its Atom-optimised kernel or ManjaroISO, which facilitates the creation of clones with all or only the packages you want.

We've left the most interesting part for last: Puppy. The Woof Puppy Builder (<http://bkhome.org/woof>) can build a Puppy from the binary packages of any other distro. This has already produced a Barebone Puppy, which is said to run even on 200MHz CPUs with 64MB RAM, plus too many, wildly different other clones to list here.

Verdict

Absolute
★★★★★
Crunchbang
★★★★★
Elive
★★★★★
Manjaro
★★★★★
Puppy
★★★★★

» For variants, go to Manjaro. To build your own, try Woof Puppy Builder.

System administration

From hardware configuration to file management, how hard is it to manage?

If you have basic Linux admin skills and want a Linux box that different users can share for basic desktop work, don't install Puppy. This is not a critique, just a service announcement.

Puppy is very well done, but mainly conceived for live usage off a CD or USB key, by single users. Some Puppy configuration tools (such as the *Menu Manager* or the *Application Chooser*) are easy and good looking. Others, like the *Firewall configurator*, are effective but archaic. But the real issue is that Puppy does not really want multiple, normal user accounts. By default, you

are root – it's your computer, so you should always be able to do whatever you want to it. Browsers and other apps that must expose themselves online run as the user spot, with a non-writeable home directory and limited privileges. When spot is not enough, you can still become fido, a normal account like you find in any other distro.

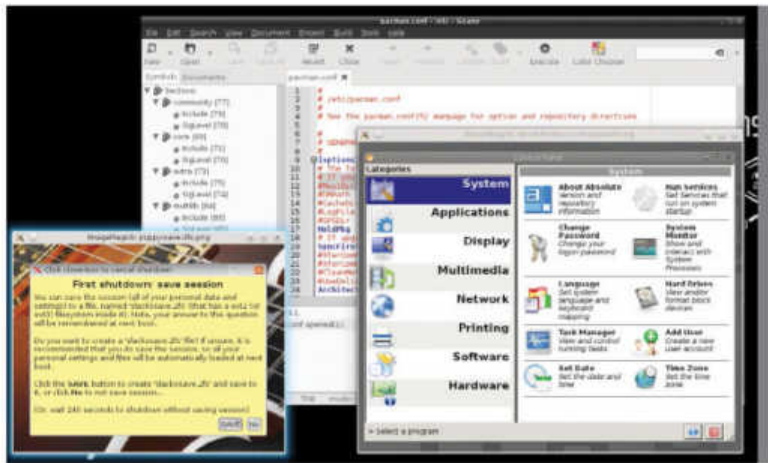
Besides, in order to boot and run as fast as possible, Puppy always tries to stay entirely in memory. Kernel, libraries, binaries, configuration files – everything works that way, without even looking for a swap partition. End user files, together

with software installed or updated after first boot, all go into one special file with a .3fs extension, on the host system or in external storage. Your main Puppy admin skill may be as simple as 'always back up your .3fs files'!

Next to Puppy, the other four distros are much more like 'ordinary' Linux. You may need the command line for certain tasks in Absolute and Crunchbang, unless you install extra packages. Absolute, for example, has a simple graphical front-end for adding users, but to remove any you should use the normal `userdel` command.

Elive has a nice mix of *Internet Configurator*, *Model Dialer* and *3G Mobile Phone* interface but not much in the 'GUI for system administration' department. The risk is that you'll spend so much time experimenting with the addictive Enlightenment control panel that you neglect anything else. If you must work at the console, Elive makes it easier with the powerful Terminology emulator.

Crunchbang and Manjaro use the Turbulence configuration interface, at least in the OpenBox versions. Manjaro also includes *MHWD* to install non-free graphics drivers. Apart from that, they are similar to their ancestor, Arch Linux.



► Puppy's .3fs saver, Manjaro's Pacman configurator, and Absolute Control Panel.

Verdict

Absolute
★★★★★
Crunchbang
★★★★★
Elive
★★★★★
Manjaro
★★★★★
Puppy
★★★★★

» GUI-based config tools (but maybe not ones you expect) are available in all.

Installing and managing apps

What if you need a program that is not installed?

No matter how good it might look, a distro is worthless if there is no way to install new programs, or update existing ones, without being a real programmer and/or fighting dependency hell.

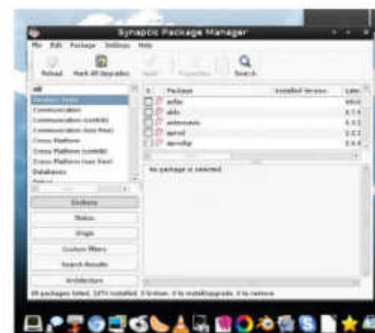
Elive is the easiest to deal with – it basically uses the same mechanisms, online repositories and graphical front-ends available in Debian and its derivatives. Crunchbang has menu entries such as 'Install *LibreOffice*'. Clicking this opens a terminal in which a script asks you to confirm, then proceeds to download and compile the sources, and install the result.

In Absolute, crude dialogs ask, for example, if you want to download the

'Multimedia' packages, that is codecs and similar libraries not included by default. Another tool will list the installed packages and offer to remove them, or install new ones previously downloaded in TXZ format.

Manjaro can find and download fully tested packages in online repositories such as AUR (Arch1 User Repository), with interfaces like *Pamac*, which can compile and install from source archives with a double-click too.

Puppy works differently, on-screen and behind the scenes. Each version gives access to a different combination of online repositories. Also, the first time you launch a program, Puppy checks if a newer version is available online and



► Elive uses the Synaptic interface, as on Debian, Ubuntu and other distros.

offers to install it. In all cases, new software is installed in the .3fs file described in the section above.

Verdict

Absolute
★★★★★
Crunchbang
★★★★★
Elive
★★★★★
Manjaro
★★★★★
Puppy
★★★★★

» Elive offers the easiest app management; it uses standard Debian tools.

Low-resource distros

The verdict

The goal of this roundup was to help you find a Linux distro that runs as fast as possible even on old computers, while still being usable by people with little Linux experience. This distro should support both office productivity jobs (text, spreadsheets and presentations) and popular web services, from home banking to online video. Ideally, it should do so through the same apps used for such activities on mainstream Linux desktops, if not on Windows, or using alternatives with comparable features and user friendliness. The system should satisfy these requirements right after install, or by installing packages from the internet with more or less the same skills necessary to perform the same tasks on more popular Linux systems. Oh, and it should also look good, if possible.

Puppy can do more or less all those things, is fast and really small, too (the version tested is a 168MB download). To tell the truth, Puppy is also such a great

package, engineering-wise, that we were really tempted to declare it the winner. We can't, though, for one reason: it is not multi-user, at least not in a way that would easily allow Linux novices to share one computer.

Absolute Linux could be perfect for people who need only an office suite and a modern browser, especially if they prefer vintage interfaces. The bare default styling of Manjaro and Crunchbang may be even more intimidating than the ancient-looking icons and menus of Puppy and Absolute. This issue, however, can be easily fixed, and both systems make software installation and upgrades easier than the other two.

Online galleries of Enlightenment screenshots show how easy it is to misuse its flexibility, making it so baroque that getting things done becomes harder. That said, Elive is fast even with its default animations, and turning them off is easy. Elive's



» **Elive makes it easy to love and use Linux, even if all you have is a five-year-old (or even older) computer.**

Enlightenment configuration is a very good balance between performance, ease of use and visual appeal. On top of that, software management is done with the same GUI tools already familiar to many Linux users. So let's declare Elive the winner, even if it is the biggest download here (about 2GB).

“Elive offers a balance of performance, ease of use and visual appeal”

1st

Elive ★★★★★

Web: www.elivecd.org Licence: N/A Version: 2.2.1 Alpha Hybrid

» Fast, good looking, and easy to manage with common Linux tools.

4th

Puppy Linux ★★★★★

Web: <http://puppylinux.org> Licence: N/A Version: Slacko 5.7 NO-pae

» For very old or single-user computers, it could well be the best solution.

2nd

Crunchbang ★★★★★

Web: <http://crunchbang.org> Licence: N/A Version: 11-20130506

» A great window manager and a more flexible browser than Manjaro.

5th

Absolute Linux ★★★★★

Web: www.absolutelinux.org Licence: N/A Version: 14.05 large

» Ideal if all you need is a browser and office suite; a bit limited otherwise.

3rd

Manjaro ★★★★★

Web: <http://manjaro.org> Licence: N/A Version: 0.8.9 OpenBox

» Almost on a par with Crunchbang, with more OpenBox config tools.

Over to you...

Did we overlook your favourite little low-resources distro? Email your opinions to ixf.letters@futurenet.com

Also consider...

This roundup isn't just for those who need to keep old hardware up and running. Even if you're satisfied with the performance of your current Linux desktop, why not try alternatives that may be even faster?

You should try Puppy Linux at least once, to get a taste of what Linux desktop development may have been. However, while Puppy is in a

world of its own, the other distros here have one thing in common: most of their speed gains over the vanilla versions of both their ancestors (Slackware, Arch Linux and Debian) and other common distros come from not using any form of Gnome or KDE.

So the lesson to take home is this: if your computer is slower than you can tolerate, don't

change distro. Install and tweak OpenBox, Enlightenment or IceWM instead. If that isn't enough, turn off file indexers and similar resource-hungry daemons. If that still isn't enough, try what is advertised as a 'low-resource' derivative of the distro that you are already using. Making Linux run faster could be much easier than you may think!

TOP 100 LINUX TOOLS

Take a stroll through the open source garden as we pick the best apps, tools and utilities available to all Linux kind.

 **With 70 Raspberry Pi top apps!**

We all have our favourite open source apps that work for us better than any available alternative.

But take a moment and step back from the *Emacs* vs *vim* type battles raging on in the Linux-verse and marvel at the sheer number of apps at our disposal. Your distros' software repositories give you access to thousands of apps, and you can install everything from fully featured app suites to nifty command-line utilities literally with the touch of a button.

There are open source apps and tools for all kinds of applications today. There's hardly any use case that isn't catered for by a community contributed app. Many of these apps have proved their mettle and offer features and performance benefits that surpass their proprietary counterparts. They have also

proved themselves to be invaluable to home and business users in more than one sense of the word. According to rough estimates on www.openhub.net, some popular apps such as *LibreOffice*, *Firefox* and *Apache* would take several hundred person-years to develop and cost millions of pounds. Yet they are all available to you for no-cost.

"Many of these apps have proved their mettle and surpass their proprietary counterparts."

Open source apps come in many shapes and sizes and you can grade them based on their usability. There are feature-rich apps, task-oriented app suites, well put-together tools, and newfangled novelty apps and games.

Some ship with well-designed graphical interfaces and others show their more versatile sides when operated from the command-line.

In this feature, we traverse this diverse and vast collection of open source gems on offer and pick the ones that are at the top of their game. In this list of the 100 best apps we've covered a wide range of categories. Whether you are a business owner, an educational institution, a developer, a home user, or a gamer, we've got something for everyone. While you'll be familiar with some of the most popular tools in this list, rest assured there are quite a few that might have missed your attention. If you've been unable to escape the clutches of commercial software, we're sure you'll find quite a few tools on this list that are suitable replacements.

Essential apps

A Linux desktop isn't complete without them.

LibreOffice

Forked from *OpenOffice.org*, *LibreOffice* has become one of the most popular office productivity suites. It includes programs for word processing, and can create spreadsheets, slide shows, diagrams and drawings, maintain databases, and compose mathematical formulae. It also offers good compatibility with documents in proprietary formats and has recently had a face lift. www.libreoffice.org



Wine

Despite the increasing number of cross-platform apps that work on Linux, there are some that still only support Windows. This includes big third-party proprietary apps, such as *Adobe Photoshop* or just small niche home-grown tools that you can't do without. For such situations, you can use *Wine*, which generally run these Windows-only apps and games with ease. The project supports over 20,000 apps. Some work flawlessly out-of-the-box while others require minor configuration tweaks. www.winehq.org

Remmina

With *Remmina* you can access a remote computer from the comforts of your desktop. It supports the widest range of protocols and will connect to all kinds of remote desktop servers. The app is easy to use, and has enough features that make it a viable option for occasional use. <http://remmina.sourceforge.net>



Thunderbird

Another gem from the Mozilla Foundation, *Thunderbird* is one of the best email clients, being easy to setup and is brimming with features. Simple setup wizards aid syncing with popular web-based email services and it can manage multiple accounts, supports encryption and is extended through add-ons. www.mozilla.org/thunderbird

KeepassX

Trying to remember different passwords for the various services is a challenge for most humans (that don't count cards in Las Vegas for fun). You can defer this task to *KeePassX* which stores password in an encrypted database. It can fill in the password automatically and also includes a random password generator. www.keepassx.org

BleachBit

Adistro accumulates a lot of digital gunk over time. *BleachBit* helps you spring clean it and protect your privacy. It also removes temporary and other unnecessary files, and has tools to securely delete files or wipe them. <http://bleachbit.sourceforge.net>



OpenSSH

When you need to interface with a remote computer, you cannot do without *OpenSSH*. It's a family of tools that provides secure tunnelling capabilities by encrypting all traffic and includes several authentication methods, and supports all SSH protocols. www.openssh.org

Gufw

You may not be using a firewall currently, and if that's because they are difficult to set up then you need *Gufw*. It features an intuitive graphical interface for managing the inbound and outbound traffic rules for various apps and services and even individual ports. Its wizard-like graphical menus are designed especially for inexperienced users. www.gufw.org



› *Gufw* has profiles and preconfigured rules to aid inexperienced users.



VirtualBox

When *Wine* doesn't cut it you can use *VirtualBox* to run an entire Windows installation inside a virtual machine. The software is also useful for installing experimental apps that you don't want to deploy on a real computer, and for testing other OSes without exposing it to real hardware. www.virtualbox.org

Clonezilla

This is a cloning solution that's distributed as a live CD and is popular for doing bare metal backup and restoration of individual PCs. It can also deploy an image to multiple computers in a lab. Clonezilla can work with a large number of popular disks, partitions and filesystem types. www.clonezilla.org



VLC

Distros ship with a functional video player. But if you need more control, there's no beating *VLC*. It supports virtually every video and audio format out there and includes handy CLI tools for advanced users. www.videolan.org/vlc



PeaZip

PeaZip is a graphical archiving tool that can work with over 130 different types of archive files and can even create encrypted ones. It integrates with popular desktops and also has a CLI for advanced users. <http://bit.ly/PeaZipSF>



Gparted

Use *Gparted* to restructure a disk on your computer. It's available as a live CD and can also be installed inside your distro. *Gparted* can create, resize, move, delete, reformat or check partitions and supports many filesystems. www.gparted.org



ZuluCrypt

Create an encrypted disk within a file or within a non-system partition or USB disk. *ZuluCrypt* has an intuitive user interface and can be used to encrypt individual files with GPG. <http://bit.ly/zuluCrypt>



HomeBank

This is a feature-rich finance app. It can import data from other apps and bank statements in popular formats. It can also detect duplicate transactions and features dynamic reports and is easy to use for budgeting. <http://homebank.free.fr>

Internet apps

Get the best of the web with these tools.



Firefox

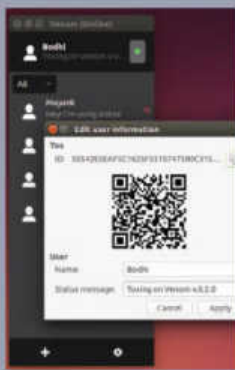
One of the most widely recognised pieces of open source software, Mozilla's *Firefox* web browser is the default browser on virtually every Linux distro. It's pretty responsive and known for its privacy features. You can customise it to the hilt and also extend it with an impressive number of extensions. www.firefox.com

gFTP

The *gFTP* client is a feature-rich client that'll get the job done, if you need to download files via FTP occasionally. It has a simple two-pane interface that shows the content of the local and remote filesystem. Using *gFTP* you can also transfer files between two remote servers. <http://gftp.seul.org>

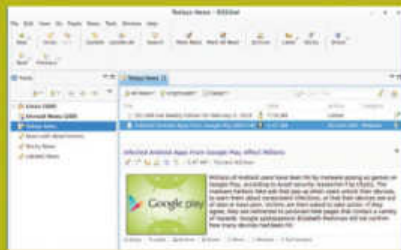
Tox

Privacy conscious users should try the new decentralised IM and VoIP client called Tox. This relies on a distributed network, which uses P2P connections, the same technology used by BitTorrent to provide a direct connection, between users for chats and, unlike other Skype alternatives, Tox uses no centralised servers or supernodes, which could be compromised. All chats are also encrypted using the peer-audited NaCl crypto library. <https://tox.im>



RSSOwl

An excellent desktop alternative to Google Reader, RSSOwl is a news aggregator for RSS and Atom News feeds that's easy to configure. The app gathers, organises, updates, and stores news in an easy to use, and saves selected items for offline viewing and sharing. www.rssowl.org



Jitsi

Jitsi is the best VoIP app, as long as you're not adverse to Java apps. It supports IM and make one-to-one audio and video calls, as well as audio conference calls. It supports many of the widely used IM and telephony protocols, including SIP, XMPP, AIM, ICQ, MSN, etc. Jitsi has all the features you'd expect from a softphone, and more, such as encrypt text chats with OTR and voice and video by establishing a ZRTP session. <https://jitsi.org>

Aria2

What makes *Aria2* a unique utility is that it can download the same file at the same time using different protocols. The lightweight CLI app can download via HTTP, FTP, BitTorrent and Metalink and can also open multiple connections to download the file faster. <http://aria2.sourceforge.net>



Midori

The go-to browser for anyone concerned about resource consumption, *Midori* is popular with lightweight distros. Despite its lightweight nature and design, Midori has all the features you'd expect from a web browser including a speed dial, tabbed interface, bookmark management and configurable web search as well as an incognito mode. www.midori-browser.org

FileZilla

For those who use FTP a lot, there's FileZilla. The client supports FTP, SFTP and FTPS protocols and has just about any configuration option you can imagine. It also has a tabbed interface so you can browse more than one server and even transfer files simultaneously between multiple servers. <https://filezilla-project.org>

Deluge

BitTorrent is popular for downloading Linux distros and there are numerous download clients. One of the best is *Deluge* which has multiple front-ends, including a graphical and a web-interface. It has features that enable advanced users to tweak it to their liking and also has a nice library of plugins. www.deluge-torrent.org



Pidgin

Pidgin is a wonderful app for instant messaging over many network protocols. You can sign in with multiple accounts in the single client and chat with many friends in different networks. You can use it to connect to AIM, MSN, Google Talk, Yahoo, Jabber, IRC and more chat networks all at once. www.pidgin.im

Games



0 A.D.

This is a real-time civilisation-building strategy game that features impressive graphics and intense battle gameplay. It's yet to have a final release but has already won accolades in its current state. <http://play0ad.com>



FreeCiv

Another strategy game that challenges players to lead their tribe 4,000B.C to the space age. www.freeciv.org



Alien Arena

A popular first person shooter with a sci-fi theme and the tournament style deathmatch of *Quake* and *Unreal Tournament*. The game has several game modes and over 60 maps, and is quite configurable. <http://red.planetarena.org>

OpenMW

OpenMW is a new game engine that recreates the popular *Morrowind* RPG. The aim of the project isn't to improve game assets or add additional features but to provide gamers a more moddable edition of the game. <https://openmw.org>



FlightGear

For fans of aircraft simulators there's *FlightGear* that aims to offer flight across real world terrain. It includes scenery for more than 20,000 airports, and can be extended with your own aircraft and locations. www.flightgear.org



Office and productivity

Enhance your workflow with these apps.



Calligra

Unless you feel you need *LibreOffice's* superior compatibility with proprietary formats, you may want to consider *Calligra*. It's a continuation of *KOffice* and unlike *LibreOffice*, *Calligra* has a modern-looking, modular design, and also uses Open Document as its native file format. It ships with a large clutch of apps. In addition to the *Words* word processor, *Tables* for spreadsheets, *Stage* for preparing presentations, and *Kexi* for managing databases, it also benefits from *Krita* for digital painting.

www.calligra.org



Zathura

This is a simple and a lightweight PDF reader that supports almost all the usual features you'd expect. You can search text strings, jump pages, zoom in and out, rotate pages, add bookmarks and more. In addition to PDFs, it can display DjVu and even encrypted documents.

<https://pwmt.org/projects/zathura>



Gnumeric

AbiWord is usually paired with the lightweight Gnumeric spreadsheet app. However, the app isn't light on features and offers a lot more functionality than proprietary spreadsheet apps. *Gnumeric* will import data from Microsoft Excel files and there are import filters for other apps as well.

www.gnumeric.org



AbiWord

The wide gap between rich text editors and word processors is occupied by *AbiWord*. It's lightweight but still offers commonly-used word processing features, which makes it a popular for lightweight distros. It also offers cloud-based collaboration capabilities via its *AbiCollab.net* service.



KMyMoney

Designed for KDE users, *KMyMoney* is a feature-rich accounting app. It supports different account types, such as Cash, Checking, Savings, etc and can categorise incomes and expenses, and can reconcile bank accounts. If your bank allows it, you can have *KMyMoney* connect to your bank directly to retrieve your account activity.

<https://kmy.money.org>



GnuCash

Home users have *GnuCash* which is similar to *KMyMoney* in terms of features, but also handles and categorises entries differently. *GnuCash* is a personal and small business accounting app that's based on double-entry for professional reporting and besides dealing with monetary transactions, it can track things such as stocks, bonds and mutual funds.



www.gnucash.org



ProjectLibre

A project management tool helps you stay on top of ongoing projects and *ProjectLibre* is one of the best. It's an award winning app that's used widely by many enterprises around the world. *ProjectLibre* has several useful features and can also visualise tasks with various charts and reports.

www.projectlibre.org



Calibre

You can use *Calibre* to manage your collection of ebooks, and supports a wide range of readers and smartphones. The app can import ebooks manually or, if you prefer, by syncing a reading device such as the Kindle. Any files imported can be sorted and grouped by metadata fields, which can be pulled from various online sources, such as www.goodreads.com.

www.calibre-ebook.com



Xournal

This app is very handy for when you need to scribble bits of information down for later. As well as typing out notes, you can use it with either a mouse or a stylus. It can also be used to add annotations to PDF files.

<http://xournal.sourceforge.net>



OpenLDAP

OpenLDAP is great for when you want to run a directory server. It implements the LDAP protocol and has all the expected features, including logging, replication, access control, user and group management etc. It also integrates with Active Directory.

www.openldap.org



Achievo

This is a web-based resource management tool with a simple interface for accessing its CRM, HRM and project management and planning tools. You can also track resources across multiple projects.

www.achievo.org



Okular

The default PDF viewer for KDE and includes a good number of useful features. Besides PDF it can also read a number of other file types, including Postscript, DjVu, CHM, XPS, ePub, TIFF, CBR, and others.

<https://okular.kde.org>



LaTeX

LaTeX is a document preparation system and document markup language based on TeX. Its purpose is to simplify TeX typesetting for documents containing mathematical formulae and is widely used in academia.

www.latex-project.org



Shutter

Besides capturing the full screen, Shutter can capture a specific area, or a window. You can also upload to a hosting service.

www.shutter-project.org



ClamAV

While most viruses and trojans will have no effect on Linux, you still can have infected files in your distro that can wreck havoc when accessed on a Windows machine. So be a good admin and use *ClamAV* to scan files.

www.clamav.net



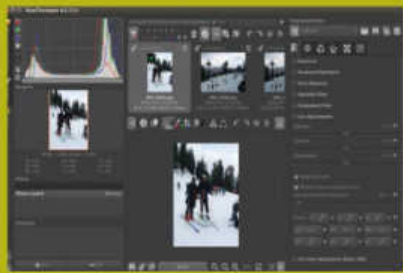
Hobbyist

Follow your passion.

RawTherapee

Do you shoot with a digital single lens reflex camera (DSLR)? Then take a look at *RawTherapee* which includes a wide range of tools for processing and converting RAW files. In addition to basic manipulations, the app has extensive options for working with RAW files. Using the app you can adjust the colour and brightness values of your images, correct white balance, adjust tones, and a lot more. Besides RAW files you can also use *RawTherapee* for editing traditional image files, and it also includes Adobe Lens Correction profiles.

www.rawtherapee.com



Scribus

A comprehensive desktop publishing program.

Scribus can be used to create professional press-ready online and print documents including brochures, booklets, books and magazines. It has a feature-rich interface and has features, such as PostScript colour separations, support for CMYK and spot colours, ICC profiles, and printer marks. *Scribus* also includes a variety of templates and styles and you also get an array of settings and tools to precisely define and position the various layout elements you require.

www.scribus.net



Krita

Although *Krita* is part of the *Calligra* suite it needs a special mention of its own. *Krita* is a digital painting and illustration app that offers many expressive brushes, HDR painting, filters, perspective grids, painting assistants, and many other features you'd expect from such an app.

www.krita.org

Stellarium

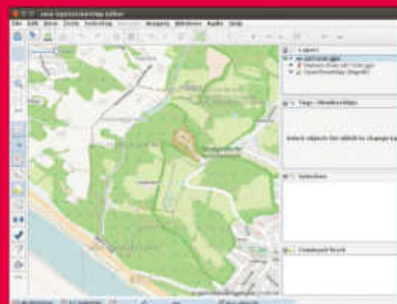
Stellarium is a free open source planetarium for your computer. It calculates the positions of the Sun and Moon, planets and stars, and draws the sky as per the users location and time. It can also draw the constellations and simulate astronomical phenomena such as meteor showers, and eclipses.

www.stellarium.org

JOSM

Keen to contribute to the mapping project, OpenStreetMap? Then use *JOSM*. It's a Java-based offline map editor that can help you plot GPS traces. You can load GPS track-logs into *JOSM* and start adding streets to OpenStreetMap instantly. Although OpenStreetMap has several other editors available, most contributors use *JOSM* for their edits, as it lets them upload changes back to OSM quickly and easily enough. *JOSM* offers several features and can be extended with plugins and styles.

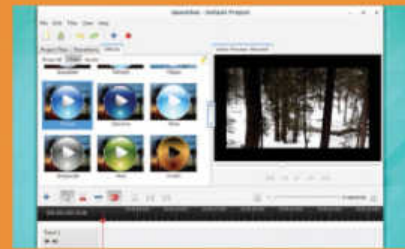
<https://josm.openstreetmap.de>



OpenShot

There are only a handful of video editors for Linux and *OpenShot* offers the best combination of features and ease of use for the home user. You can use it to combine videos, audio tracks, and still images together and add in captions, transitions, and more, and export the final product in a variety of formats. *OpenShot* can also use *Blender* to create 3D animated titles

www.openshot.org



Inkscape

Another pro-quality tool, *Inkscape* offers advanced vector graphics editing and is popular for drawing vector art, line art, and designing logos and graphics. It's brimming with features, such as markers, clones, alpha blending and more, and is often compared to expensive proprietary apps such as *Illustrator* and *CorelDraw*.

www.inkscape.org

Cinelerra

Cinelerra is excellent if you need to edit more than home videos, as it's the most advanced non-linear video editor and compositor for Linux. It supports HiFi video and audio and is resolution and frame-rate independent, which enables it to edit videos of any size. The app has several advanced features, such as overlays, denoising, normalisation, time stretching, color balance, compositing, real time effects and a lot more. It also includes a compositing engine for performing tasks such as keying.

www.cinelerra.org

Media

Comix

Digital comics are distributed as comic book archive files that mainly consist of a series of image files, typically PNG or JPEG files, stored as a single archive file. *Comix* can read digital comics in virtually every format.

<http://bit.ly/ComixApp>

FontForge

FontForge is a feature-rich app for creating and editing fonts and supports all common font formats. It can extract information from a font file as well as convert from one format to another, and can be used for previews.

<http://bit.ly/FontForge>

CairoDock

CairoDock is a MacOS X dock-like app. One of its main advantages over other docks is that it doesn't require a compositing window manager to work and can add bling to older low-powered machines.

www.glx-dock.org



Audacity

If you need to work with audio, you should use the powerful *Audacity* sound editor. You can trim audio, combine tracks, and even stack multiple tracks, as well as export to a number of formats and quality settings.

<http://bit.ly/AudacityApp>

MPD

The Music Player Daemon is an audio player with a server-client architecture, which means you can control it remotely from another computer. It plays audio files, organises playlists and can maintain a music database.

www.musicpd.org

Development

Power tools and programs for power users.

jEdit

This is a text editor for programmers that supports auto indent, and syntax highlighting for more than 140 different programming languages. The app enables you to define complex macros and offers a powerful and user-friendly keyboard mapping system. It's highly configurable and customisable, and you can extend its functionality by adding plugins.

www.jedit.org



Eclipse

There's no beating *Eclipse*, the most feature-rich IDE. Although Java is its speciality, Eclipse supports a range of languages via plugins. In fact, its plugin marketplace is an indispensable resource. Eclipse does code refactoring and you can use it to extract the selection as a local variable or method. Since it can target multi-person installs, it handles version control very maturely

www.eclipse.org

BlueFish

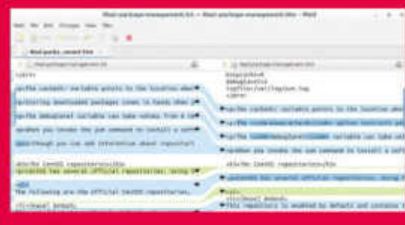
Do you develop for the web? *Bluefish* is a multi-language editor that's designed for web developers. It supports many programming and markup languages and focuses on dynamic and interactive websites. It supports code block folding, unlimited undo/redo, automatic tag closing, and syntax highlighting. Another useful feature is the snippets bar from where you can add the most common snippets of code for a variety of languages. *Bluefish* also has support for popular open source web apps such as MediaWiki and Wordpress.

<http://bluefish.openoffice.nl>

Meld

A graphical diff tool, *Meld* enables you to compare two or three files as well as whole directories. It includes features, such as syntax highlighting and direct file editing, and using the tool you can easily isolate and merge the differences. *Meld* can also be used to browse various popular version control systems such as *CVS* and *Subversion*.

www.meldmerge.org



KompoZer

New and experienced HTML programmers will save a lot of amount of time and effort with the *KompoZer* editor. It has an intuitive interface and includes a colour picker, an FTP site manager, CSS editor, customisable toolbars, forms, spell checker, markup cleaner and can also validate code using W3C's HTML validator.

www.kompozer.net



Gimp

Despite its name, Gimp is a powerful, comprehensive image manipulation program. It offers a wide range of tools for professional-quality photo retouching and image manipulation capabilities for free. It also offers a huge list of features and supports all the common graphics file formats.

www.gimp.org



Blender

With *Blender* animators can create 3D printed models, visual effects, art, interactive 3D applications and video games. The app provides a wide range of features that can be used to create 3D animation films. It's a one-stop 3D package and includes a gaming engine, a video sequence editor, production-ready camera and object tracking, a large library of extensions, and an advanced physics engine. It can render fluid dynamics and simulate the movement of elastic objects and clothes.

www.blender.org

Geany

You don't need a full-blown IDE if you only program occasionally, which makes *Geany* a good choice. It's a cross between a plain text editor and an IDE with support for the popular languages and nifty features like a compile/run button, a listing of functions defined in the currently opened file, and much more.

www.geany.org

APTonCD

Suddenly realise that you need to move your Ubuntu installation or need to give a friend a copy of your setup? With *APTonCD* Ubuntu users can back up all of their installed packages to an ISO image, which can then be added as a software source on another installation. You can use this source to restore the packages on to the system or keep everything in the APT cache.

aptoncd.sourceforge.net



Clementine

Use *Clementine* to play locally stored music and streaming audio. The app has an attractive interface and it also helps organise and transfer music to various devices, and integrates well with popular cloud services.

www.clementine-player.org

Icecast

With *Icecast* you can stream music across the network. *Icecast* supports many audio streams simultaneously and listeners can access a stream via a remote media player and also configure MPD as a source.

www.icecast.org



Amarok

If you use KDE your distro may already include this music player, *Amarok*. It too integrates with several online audio services, and its features include creating dynamic playlists, bookmarks, scripting, context view.

<http://amarok.kde.org>

LMMS

LMMS is digital audio workstation that produces music by synthesising sounds, arranging samples, and playing them on a MIDI keyboard. It also has a song editor and plugins to simulate instruments and effects.

www.lmms.io



Kodi

Until recently *Kodi* was known as *XBMC*. It's an excellent option for users who wish to turn their PCs into media hubs. It plays most kinds of media files and works with TVs, IR and bluetooth remote controls.

www.kodi.tv

Utilities

Apps that let you do more with your computer.

Gnome Tweak Tool

Not satisfied with the stock Gnome desktop? Use the *Gnome Tweak Tool* to customise several aspects, including the appearance settings of the desktop. With this tweak app you can also change the behaviour of the Windows and Workspaces, manage extensions and you can even circumvent the design philosophy of Gnome 3 by placing icons, files and folders on the Gnome desktop.

<http://bit.ly/GnomeTweakTool>



digiKam

One of the best photo management tools for Linux is *digiKam* and it has features that'll appeal to all kinds of users. It recognises all major image file formats and can organise and sort images based on metadata. The app also has plugins to export images to various online services.

www.digikam.org



K3b

Although it's designed for KDE, the *K3b* optical media burning utility is one of the finest for the job. The app can burn multiple El Torito boot images, audio CDs, VCDs, SVCDs, mixed-mode CDs, eMovix CDs, and DVDs. It can also rip DVDs and write ISO images.

www.k3b.org

Grub Customizer

Grub 2 is the most popular Linux bootloader that's used by virtually all major distributions. It's an impressive piece of software with lots of options. The *Grub Customizer* is a simple to use graphical tool, which enables you to quickly customise all aspects of the bootloader, including its appearance.

www.launchpad.net/grub-customizer



DOSBox

Relive the good ol' days with *DOSBox* and play your favourite classic DOS games that won't run on your modern hardware. This is an x86 PC emulator that creates an IBM PC compatible computer complete with compatible graphics and sound cards. The app can also simulate networking hardware for multiplayer games on the local network and even over the Internet. The *Wine* project even uses code from *DOSBox* to bolster support for DOS apps.

www.dosbox.com

Avidemux

Avidemux is a video editor and converter that can be used for basic cutting, filtering and encoding tasks. It supports many file types, including AVI, MPEG, and MP4. The app is designed for users who know what they want to do but also provides an intuitive interface so that tasks such as cutting and appending videos are pretty straightforward. The app has some presets and users can also save custom settings that make the app easier for new users to operate.

<http://fixounet.free.fr/avidemux>



Handbrake

When the need to convert a video arises, *Handbrake*, the video transcoder app does a commendable job. It can convert nearly any format and supports a wide range of video codecs. One of its best features is built-in device profiles for popular devices that make the conversion process easier.

www.handbrake.fr

EasyStroke

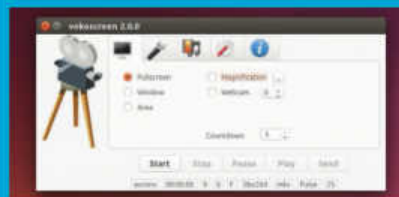
Want to control your PC with the flick of the mouse? The *EasyStroke* app lets you define and manage gestures by recording the movements of your pointing device while holding down a specific mouse button. You can then configure actions that'll be executed when the app recognises the defined stroke.

<https://easystroke.sourceforge.net>

Vokoscreen

A feature-rich screencasting app worthy of note is *Vokoscreen*, which is based on FFmpeg for handling multimedia data. *Vokoscreen* can capture both video and audio, with options to record the entire screen, window or a selected region, along with video from a webcam. The app supports MPEG4, x264, MP3 and Vorbis codecs and can save files in either .AVI and .MKV containers. The app offers some controls such as the ability to change the video quality and frames captured per second and can be used to make screencasts of games.

www.kohaupt-online.de/hp



Terminal

Ncmpcpp

This is a command-line MPD client that's easy to use and customisable. It provides useful features such as the ability to sort playlists, song lyrics, item filtering, fetching artist's info from last.fm, tag editor and much more.

<http://bit.ly/Ncmpcpp>



Samba

Samba is a suite of programs that enables Linux users to access and use files, printers and other commonly shared resources on a Windows PC on a network and does this by supporting the SMB protocol which.

www.samba.org

rTorrent

Here we have a command-line BitTorrent client with an ncurses interface. You can run it as a daemon and manage it with *screen* and since it supports SSH you can manage your torrents from any remote machine.

<http://bit.ly/rTorrent>

Links2

There are lightweight browsers and then there's *Links2*. This is a web browser that can render complex pages and even has a pull-down menu. It's also special because it's a CLI browser that you operate via the keyboard.

<http://links.twibright.com>

Midnight Commander

Before the days of graphical file managers, real hackers used *Midnight Commander*, known as *mc*. It's still your best option if you regularly find yourself in the console environment a lot.

<http://bit.ly/MidnightCdr>

Admin tools

Take charge of your distro with these power apps.

Redo Backup

We've mentioned the *Clonezilla* cloning solution earlier in the feature, but if all you need is a tool to swap out an old disk for a new one, then you use *Redo Backup and Recovery*. The tool is designed for inexperienced users and has the simplest of interfaces.

www.redobackup.org



XAMPP

The XAMPP stack gives you a single package that you can use as a sandbox to test and develop web apps. It includes all the necessary components such as *Apache*, *MySQL*, *PHP*, and *Perl* as well as several other libraries, modules and tools, such as *phpMyAdmin* and *FileZilla* for managing the stack components. Once installed, you can manage the various services via a graphical control panel.

www.apachefriends.org

Déjà Dup

The app's minimal GUI sets itself apart from the various other backup apps you'll find, and it lets you configure backups within a matter of minutes. *Déjà Dup* is based on *Duplicity* and provides just the right number of features for desktop users who aren't used to the ways of a backup tool.

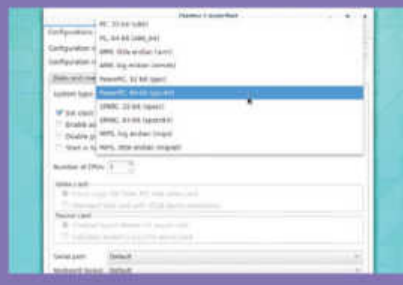
<http://live.gnome.org/DejaDup>



Qemu

It's a feature-rich multi-purpose processor emulator and virtualiser. You can use it to create virtual machines and even emulate various hardware architectures. If you have the right hardware on tap (a processor with hardware virtualisation extensions), you can use *Qemu* with *KVM* in order to run virtual machines at near-native speed.

www.qemu.org



Mondo Rescue

Mondo is a unique backup solution that creates bootable backup and restoration disks customised for the system being used. *Mondo* has a text-driven interface and works with a wide range of file systems and can use a variety of media as backup mediums.

www.mondorescue.org



Open Media Vault

When you need more protection for your data than a simple backup then you need to deploy a NAS server. The *Open Media Vault* project is a Debian-based server that offers the power of commercial options in a way that's easy to setup and manage.

www.openmediavault.org

Conky

Concerned about the resource utilisation on your PC? *Conky* is a nifty little app that lets you keep an eye on your system. It can monitor and report on the states of various components. The tool is very flexible and highly configurable and can also display information from apps, such as weather updates.

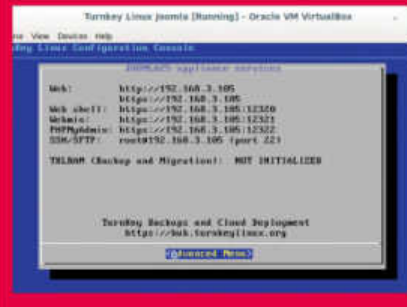
<http://conky.sourceforge.net>



Turnkey Linux

The Turnkey project produces appliances which you can use to deploy a new server in a jiffy. A Turnkey appliance is a self-contained system that packs in a fully functional web app that runs on top of Just enough Operating System (JeOS) components required to power that particular app. All the appliances are based on Debian but are available in several formats depending on the hardware that you want to deploy it on. Once they're up and running you can manage each appliance using a browser-based interface.

www.turnkeylinux.org



Zentyal

The Zentyal distro has all the components you need to run a gateway server. The distro simplifies the process of setting up, monitoring and controlling the components of the server with a host of custom management tools and helps you configure the servers without mucking about with config files.

www.zentyal.org

Mutt

Mutt is to email what *Links2* is to the web browser. It's a text-based mail client that is highly configurable and it supports both POP and IMAP protocols and has all the usual features you'd want from an email client.

www.mutt.org



Profanity

Profanity is a console-based client for the XMPP protocol that supports multi-user chats and OTR message encryption.

www.profanity.im



Canto

Want to do more from the command-line? Get the *Canto* CLI RSS feed reader. It supports RSS, Atom and RDF feeds and imports and exports feeds in OPML format. It has lots of customisation and even configure it with Python.

<http://bit.ly/CantoRSS>



mpg123

This is an MP3 audio player for the command-line that supports gapless playback. It's so good that its decoding library, *libmpg123* is used by other audio players for MP3 playback.

www.mpg123.de



FFmpeg

One of the most versatile media conversion utilities, *FFmpeg* can manipulate virtually any type of media file in various ways, such as changing bitrate, extract audio, record streams, extract stream and much more.

www.ffmpeg.org





Low resource applications

Got an old PC? Then we offer you this smorgasbord of lightweight applications that won't eat up all your memory.

We've covered a few lightweight Linux distros (see page 84), but the plethora of pint-sized applications out there deserves attention, too. For the vast majority of all your other Linuxing, there are some small yet perfectly formed programs that do everything you need, and do so without pulling in a truckload of dependencies and eating all your memory.

If you're really serious about resource usage, you should consider using console applications. This can be daunting at first, but once you learn

the requisite quota of magic keystrokes and break out of the clicking-and-dragging mindset, you'll be surprised how much more efficient you can be.

Furthermore, in so doing you'll look much cooler. If you still don't want to go down that road, there are plenty of slimline graphical applications worth considering.

File managers

Files are important, so a functional file manager is vital. Our first suggestion, then, is somewhat controversial – don't bother with one. Or, more

precisely, be your own file manager. With some basic shell-fu, you can do all manner of balletic moving and systematic renaming, and with tab-completion you can obviate the problem of typing long filenames.

If you do crave a purpose-built file manager, but are happy with a terminal-based one, then your first stop should be *Midnight Commander* – sorry, *GNU Midnight Commander*. This is an Orthodox File Manager (OFM) offering a dual pane view with a command terminal in the style of the classic *Norton Commander*. It's pretty intuitive: [Tab] moves between panes, and the

The Suckless and KISS philosophies

Growing frustrated with the seemingly inexorable bloat of core tools and libraries, a community of motivated hackers have established <http://suckless.org>. This collective aims to produce software based on the key principles of simplicity, clarity and frugality – “software that sucks less”, to use their own words. The project is geared towards more advanced users, because this group is generally the most frustrated by the aforementioned symptoms. The collective’s

efforts to date include, but are far from limited to: the *dwm* window manager, the *dmenu* menuing system, the *Surf* web browser and *st*, a simple terminal. On their website you’ll find a list of software that “rocks”, some of which is included in this article. Also, you’ll find lists of software, and assumptions made by software, that are deemed to “suck”.

The more general KISS (Keep It Simple, Stupid) idealogy, oft-uttered on the Arch Linux

wiki but apropos to so many situations in life, aims for simplicity, eschews unnecessary complexity and strives for elegance through minimality. In terms of software, this speaks to the underlying UNIX ethos of small tools that do one thing and do it well. One can equally see it encompassing those applications (particularly graphical ones) that avoid over-worked interfaces, unnecessary features and pull in myriad software and library dependencies.

actions performed by Function keys are helpfully shown at the bottom. You can even use the mouse and open files in their native applications (via *xdg-open*) if you run it in an X session. *Midnight Commander* also has a native viewer and editor (hex and ascii), can open archives natively, can browse FTP sites and windows shares, and comes with some lovely themes. If you want something a little more advanced, then direct your attention to *vifm*, which features ‘vi-like keybindings’. As such, we shall say no more about it.

Moving on to graphical species, we’ll begin with *SpaceFM*. This offers a multi-panel tabbed interface and its own virtual filesystem, and it is highly customisable beyond recognition. It can be built for *GTK+2* or *3* and works seamlessly with the dependency-light *Udev* device-mounting program, so you’re spared asking yourself, “What is this, 2004?” when you’re prompted for the root password in order to mount a USB stick. *SpaceFM* can even manage desktop icons, which may appeal to you if you’re using a standalone window manager, or may be irrelevant if you consider desktop icons to be a relic from 1995.

SpaceFM was forked from the venerable *PcManFM* back in the *HAL* days. The latter is the default file manager in Raspbian and LXDE and differs mainly in its reliance on the *Gnome VFS* (and hence the *Udisks2* framework) for mounting of drives, shares and whatnot. If you don’t mind installing a bunch of Gnome 2 libs and want an OFM, then *Gnome Commander* is probably a worthy candidate. If you are a KDE person with similar desires, then try *Krusader*.

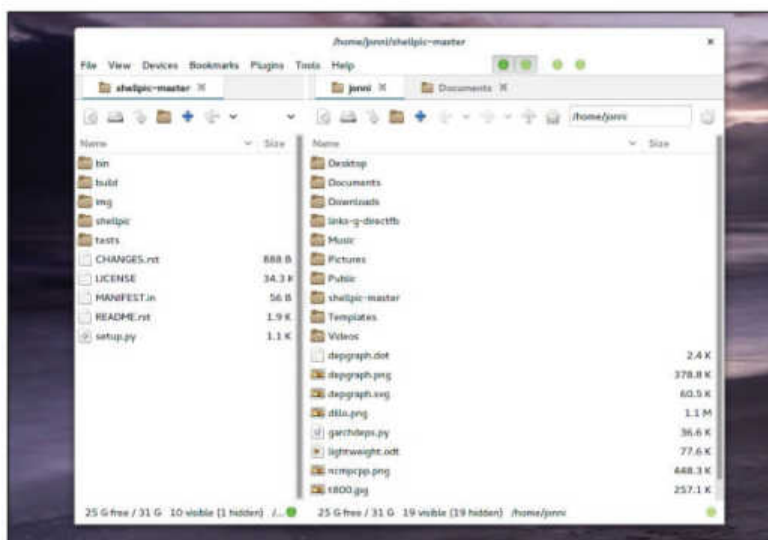
Terminal emulators/multiplexers

If you’re using a desktop environment, then chances are you’re satisfied with the terminal program provided therein. But there are alternatives, some of them rather good.

Of particular note is *rxvt-unicode*, which builds on the venerable *Rxvt* with features such as unicode support, transparency (pseudo and true) and support for *Xft* fonts. It supports Perl extensions which can provide clickable URLs and a tabbed interface, although that relies on *GTK+2*. Somewhat uniquely, it’s built around a client server model, so once the daemon is loaded, opening terminal windows is both swift and memory-light.

If you want to be really minimal, then you should use *st* (*Simple Terminal*) from <http://suckless.org>, but it might come as a surprise that there’s no scrollbar buffer, and that you have to edit a file to make the [Del] key work in *Bash*.

Next, we have the *GTK*-based *Lilyterm*, *Sakura* and *Evilvte*. *Lilyterm* works best with *GTK+2*, *Sakura* only with *3*, and *Evilvte* can work with either *2* or *3*. All three have very few dependencies besides the venerable Gnome VTE library and offer tabs, transparency and unicode support.



▶ *SpaceFM*, a lightweight file manager running in Gnome 3.

Terminals that slide down from the top of the screen à la *Quake* are popular too. These include *Yakuake* for KDE and *Guake*, which doesn’t require much besides *VTE* and the Python bindings for *GTK+*. There is also *Tilda* (the tilde key summoned the terminal in the original *Quake*), which doesn’t require these bindings. Having a terminal a hotkey away is pretty useful, but sometimes you do need multiple windows, so this might not be a complete solution. We should also mention *Terminator* (“the robot future of terminals”) here, since it enables the user to tile sessions in all manner of wild configurations.

Remember also that you get a whole bunch of console terminals for free too (accessible via [Ctrl]+[Alt]+[Fn] keys). If you’re stuck at the console, you can use these as primitive tabs. Or you could up your game and get a terminal multiplexer such as *tmux* or *screen*.

These are mainly used for keeping remote programs running after logout, but can do much more – multiple sessions, scrollbar buffers, copy and paste, awkward keyboard shortcuts, the list goes on.

Web browsers

Depending on what you’re doing, surfing from the console might be entirely reasonable – for example, if you simply



“If you want to be really minimal, then you should use st”



want to do a quick Wikipedia check, or search the Gentoo documentation for the USE flags required to make your login manager work. In this case, something such as *Lynx* or *Links* (in text mode) is all you need. However, many web pages don't render terribly well; *Lynx* doesn't support CSS or JavaScript, for instance, and while *Links* does support JavaScript, it might not help much in text mode. *Links* does, however, have a graphical mode which makes use of a framebuffer. In this mode frames, tables and images are supported, but don't worry – it can't do *Flash*.

The lightest web browser for X is the fltk-based *Dillo*, which manages to squeeze both substantial CSS support and tabbed browsing into its tiny footprint. Its lack of JavaScript support and history-keeping can be seen as security features. Cookies are supported, but turned off by default. Likewise basic SSL transactions, although users are warned that this support is very much in an alpha state. *Luakit* (*GTK+*) and *dwb* both use the *Webkit* engine to provide standards-

compliant rendering with thoroughly minimal interfaces. They both rely rather heavily on keyboard shortcuts and are

both thoroughly extensible. More conventional Webkit-based browsers include *Midori* (*GTK+*) and *QupZilla* (*Qt*).

“Dillo squeezes tabbed browsing into its tiny footprint”

Email clients

Mutt is a terminal-based email client that's been around since the mid-90s. It supports IMAP, SMTP, SSL, GPG and third-party plug-ins. Of course, there are some keyboard shortcuts to learn, and if you want to look at your mail offline you'll need to set up *offlineimap* (IMAP) or *getmail* (POP). *Mutt* uses the external text editor of your choosing, and you can make a sort of address book using aliases, or use the purpose-built *Abook* application. Back in 1991, the *Pine* email client was released, but source code was only available for UNIX. *Pine* was pretty easy to learn, and came bundled with a simple-but-powerful text editor called *Pico*. The *GNU Nano* editor we all know and love is a clone of this. The Apache-licensed *Alpine* (Alternatively Licensed Pine) was released in 2007 and although it is no longer under active development, a project

called *Re-alpine* sought to continue it. The latest release of *Re-alpine* was in mid-2013.

On the graphical front we have *Sylpheed* and its slightly more featureful fork *Claws Mail*, both of which use *GTK+2* and support various plug-ins. *Sylpheed* does almost everything you need and absolutely nothing more. *Claws Mail*, meanwhile, offers support for spam filtering, GPG and can render HTML mail through Webkit. If you like *Qt* and IMAP, then you should have a look at *Trojitv* which supports HTML mail (through *Webkit*) and has been designed to be as gentle as possible with respect to resource usage.

Music players

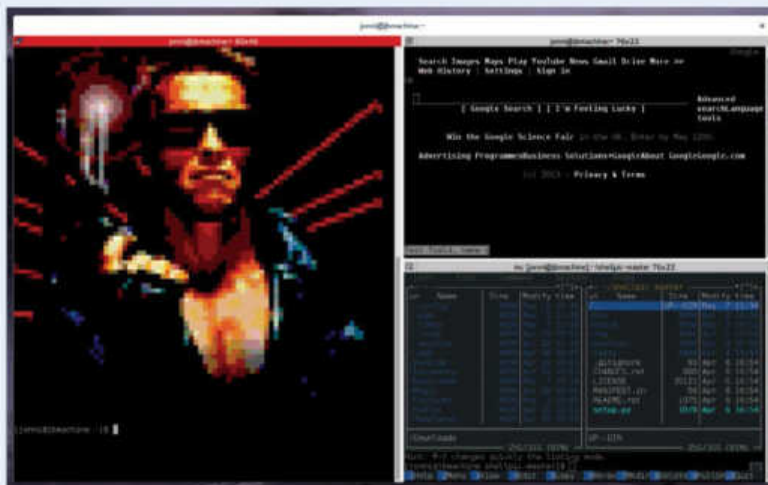
MPD (*Music Player Daemon*) is a superb program that will maintain a database of your music collection and play tracks in response to commands from a (possibly remote) client program, of which there are many. It is incredibly lightweight, which means it's possible to use it with a *ncurses*-based console client such as *ncurses* (or the even more confusingly named *ncmcpmp*) without your CPU even noticing. *MPD* isn't shy of features; it plays anything the *FFmpeg* library understands (in other words, anything). Spotify and SoundCloud playlists are supported, and all manner of plug-ins are available. If you just want a simple *ncurses* music player, then your first stop should be *Cmus* or *Herrie*.

There are graphical clients for *MPD*, too, such as *Xfmcp* for XFCE and Gimmix, but development seems to have dried up on most of the other *GTK+* ones. On the *Qt* front, all we were able to come up with was *Cantata*. In terms of standalone applications, there is the superbly-titled *Deadbeef*, which uses *GTK+2* or 3 and not much else, yet supports all manner of formats, tag editing, gapless playback and has many plug-ins available.

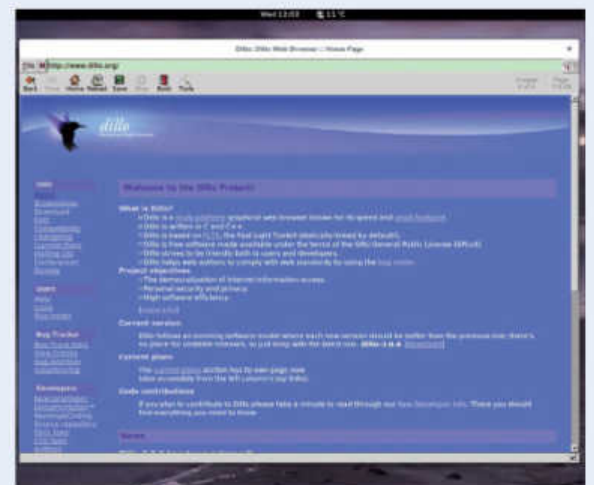
Slightly larger, but far from overweight, is the mighty *Audacious*, which you can make look like Winamp/XMMS. Similarly, for *Qt* check out *Qmmp*. Speaking of XMMS, XMMS2 is a client/server project that aims to continue the legacy of its hugely popular predecessor. Clients catering to all tastes are available: <https://xmms2.org/wiki/Clientlist>.

Image viewers and editors

You can view pictures at the console, and all you need to use is Lars Solberg's *Shellpic*. It's written in Python, only depends



► Terminator running *shellpic*, *links* and *mc*. You could have done much more governing this way, Arnold.



► The *Dillo* browser does a great job at rendering <http://dillo.org>; *YMMV* works for other sites.

GTK+ and Qt: the Bloods and Cripps of the Linux desktop

GTK+ and Qt are cross-platform widget toolkits. These enable a unified look and feel across applications with respect to menus, dialog boxes and the like. GTK+ is used by Gnome, while LXDE, Xfce and Qt are used by KDE and Razor Qt. If you want a truly minimal desktop, then ideally you would only install applications that use one particular toolkit.

On one level, GTK+ is lighter than Qt, but the situation is complicated by the fact that some

applications prefer version 2 and others version 3. Further, while there are many GTK+ themes out there, very few work well with both versions, hence the many forum posts of the genus, “my GTK{2,3} apps look awful”. Some applications work with either version, and in this case distributions tend to package their GTK+3 incarnations. If you want a GTK2+ version, then building one is usually pretty straightforward if you’re OK with making minor edits to configure

scripts. If you’re an Arch Linux user, then the Arch User Repository has a bunch of **-gtk2* packages ready made for you, saving you the indignity of 100MB of GTK3 and friends to install.

If you don’t mind having a heterogeneous GTK+ and Qt setup, then you can get a reasonable degree of consistency by using something such as *QGtkStyle* to make Qt look like your GTK+ theme, or the GTK-Qt engine for the other direction.

“MPlayer can play pretty much anything you throw at it”

on the *Pillow* library, and uses ANSI escape codes to display images in as many colours as your terminal supports. If you are working outside of an X server, but have KMS enabled (or are otherwise willing to fiddle with things such as *uvesafb* in order to get a high resolution framebuffer), then you can use *fbi* (*framebuffer image viewer*) to display images slightly more faithfully.

You can also process images with the *convert* and *mogrify* commands from *ImageMagick*. If you have a graphical environment, then the latter’s display program is about as lightweight an image viewer as you can get, although it’s entirely command-line driven. Likewise the popular *feh* program. Both of these can work with multiple files and be used to set the desktop background in standalone window manager situations. For a more traditional image viewer, you should direct your attention towards *Geeqie* – a simple GTK+2 program that can work with EXIF data and .raw images. If you want basic image-editing capabilities as well, then *Mirage* (built on *PyGTK*) and *Converseen* (Qt4) are both good candidates.

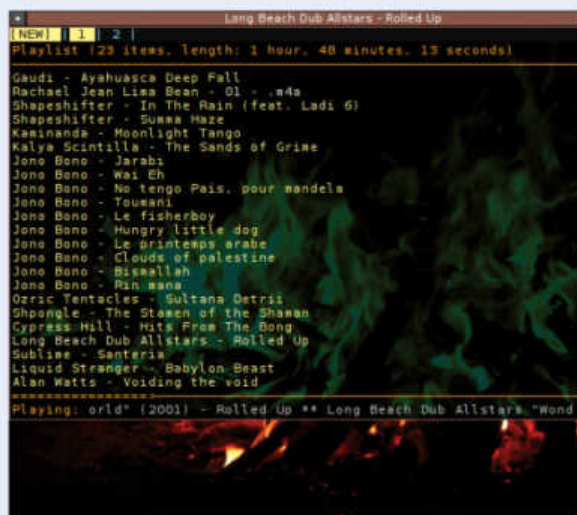
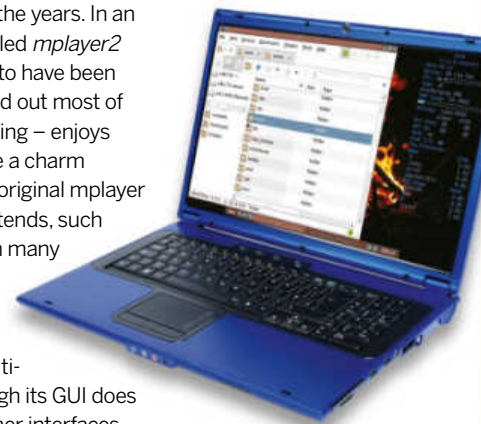
It would be remiss not to mention the stalwart *MPlayer* here – it can play pretty much anything you throw at it (and can do so on a framebuffer device, or as ascii art) and can fulfil all your encoding requirements via *mencoder*. Sadly,

despite the codebase still being actively maintained, it has become fragmented and spaghetti-like over the years. In an attempt to modernise it, the imaginatively-titled *mplayer2* fork was set up, although this project seems to have been eclipsed by the superb *MPV*, which has ripped out most of the old zombie code and – at the time of writing – enjoys rapid development. Its internal GUI works like a charm (compare it with the abomination that is the original mplayer GUI), so there’s no need for the desktop frontends, such as *Gnome-mplayer* and *smplayer* (Qt), which many considered de rigueur for its predecessor.

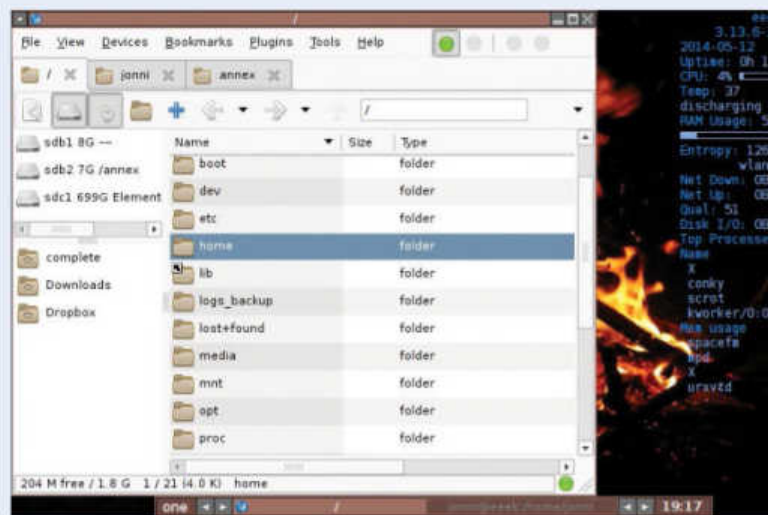
Media players

VLC, the media player of choice for most multi-format fans, remains as reliable as ever, though its GUI does rely on Qt4. That said, it has all manner of other interfaces (*ncurses*, web, remote-control) and can output to a framebuffer or as ascii art. *Snappy* is a medium-weight GTK+3 player which is built on the *GStreamer* and *Clutter* frameworks, but has few other Gnome dependencies and looks pretty.

And that concludes our summary of diminutive applications. Hopefully you’ve found something lightweight with which to replace something heavy, and have clawed back some of your precious resources. Even if you have a powerful computer, there’s much to be gained from learning the tools we’ve covered here, or others of similar ilk. They can inspire a more efficient workflow, show you new ways to get stuff done, and improve your memory through the learning of keyboard shortcuts.



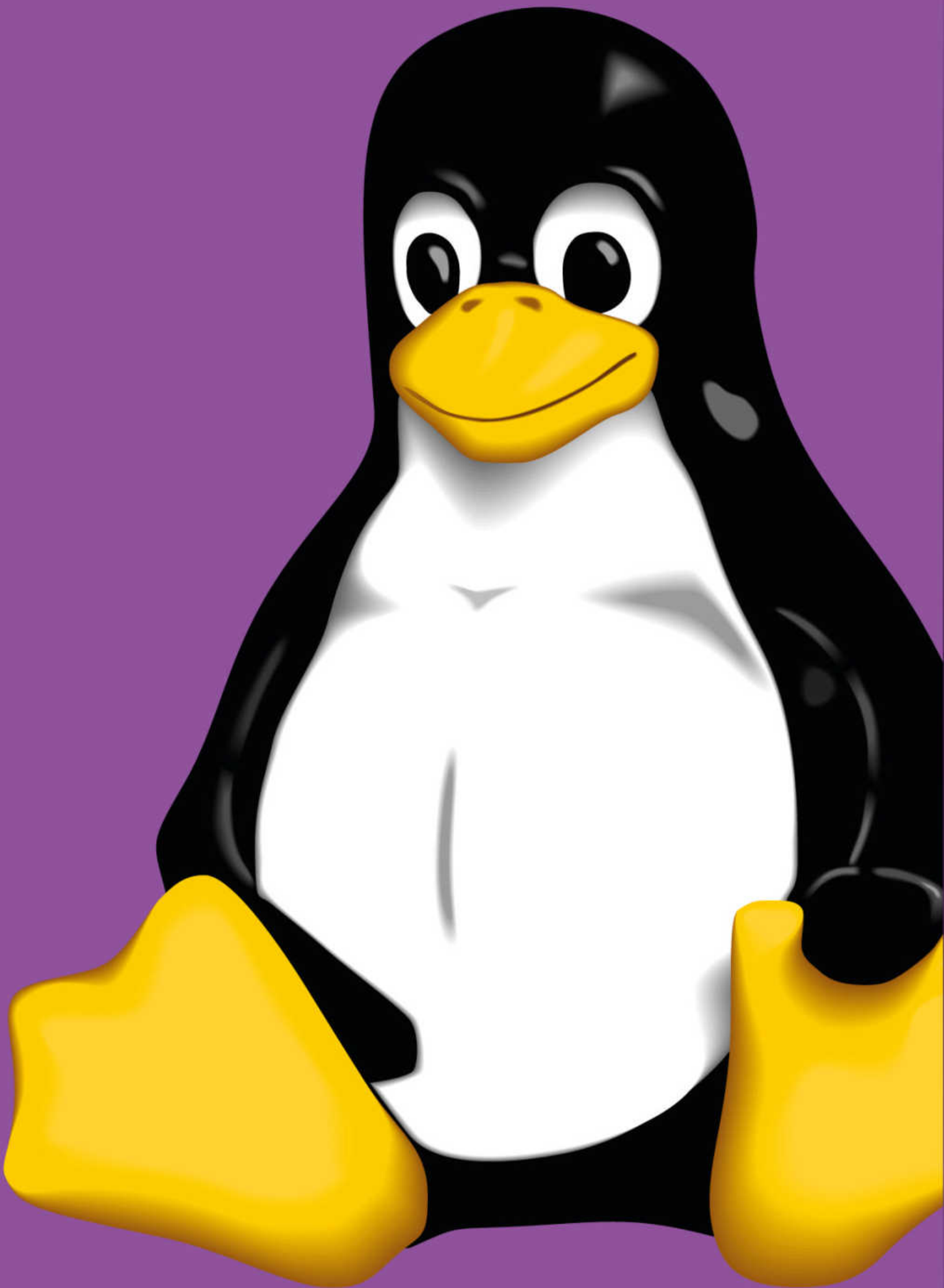
▶ *Ncmpcpp* and *Conky* running on *Fluxbox*. Sleek.



▶ *SpaceFM* compiled for GTK2+ with three tabs and a 15MB memory footprint.

In-depth

Terminal basics.....	104
Apt-get in action.....	106
Disk managment.....	108
Archiving.....	110
Core terminal programs.....	112
Terminal tricks.....	114
User accounts.....	116
Built-in help.....	118
Sysadmin basics.....	120
Super-user control.....	124



Terminal: Getting started

There's no need to be afraid of the command line – we're here to help you with your first steps into the world of text commands.

It won't be long after starting to use Linux that you ask a question and the answer begins with, "Open a terminal and..." At this point, you may be thrown into an alien environment with typed commands instead of cheery-looking icons. But the terminal is not alien, it's just different. You are used to a GUI now, but you had to learn that, and the same applies to the command line. This raises an obvious question: "I already know how to use a windowed desktop, why must I learn something different?" You don't have to use the command line, almost anything you need can be done in the GUI, but the terminal has some advantages.

It is consistent The commands are generally the same on each distribution while desktops vary.

It is fast When you know what you are doing, the shell is much faster for many tasks.

It is repeatable Running the same task again is almost instant – no need to retrace all your steps.

There is more feedback Error messages from the program are displayed in the terminal.

Help is available Most commands provide a summary of their options, while man pages go into more detail.

You can't argue with the plus points, but what about the cons? Well, apart from not giving us pretty screenshots to brighten up the pages, the main disadvantage of the terminal is that you need to have an idea of the command you want to run, whereas you can browse the menus of a desktop system to find what you're after.

In this tutorial, we will look at the layout of the filesystem on Linux, and the various commands that you can use to manipulate it. On the following pages we will cover several other aspects of administering and using a Linux system from the command line.

What ls tells us about files

```

-rw-r--r-- 1 nelz users 1.3K Feb 14 12:46 FirefoxOS_emulation.
-rw-r--r-- 1 nelz users 350K Feb 12 10:53 FirefoxOS_emulat.png
-rw-r--r-- 1 nelz users 3 Feb 14 13:12 .issue
-rwxr-xr-x 1 nelz users 2K Jul 19 2013 lxfanswers
-rw-r--r-- 1 nelz users 1.1K Feb 14 12:46 Misty_MINT.txt
-rw-r--r-- 1 nelz users 523K Feb 28 12:44 New_laptop_no_D.png
-rw-r--r-- 1 nelz users 3.4K Feb 14 12:41 New_laptop_no_DVD
-rw-r--r-- 1 nelz users 72K Feb 14 12:50 Old_but_Smart.png
-rw-r--r-- 1 nelz users 2.9K Feb 14 12:02 Old_but_Smart.txt
-rw-r--r-- 1 nelz users 124 Feb 14 13:13 .order
-rw-r--r-- 1 nelz users 1.9K Feb 13 11:16 QR.txt
-rw-r--r-- 1 nelz users 2.8K Feb 27 20:48 Raspberry_capture.tx
drwxr-xr-x 2 nelz users 2 Mar 5 15:52 temp
-rw-r--r-- 1 nelz users 111K Feb 16 23:18 tut_1.png
-rw-r--r-- 1 nelz users 79K Feb 14 23:45 tut_2.png
-rw-r--r-- 1 nelz users 583K Feb 16 23:14 tut_3.png
-rw-r--r-- 1 nelz users 7.8K Feb 28 12:47 Tutorial183.txt
-rw-r--r-- 1 nelz users 8.1K Feb 14 23:31 tutorial.ncd
-rw-r--r-- 1 nelz users 7.3K Mar 4 11:22 Tutorial.txt
-rw-r--r-- 1 nelz users 68K Feb 10 11:49 Very_little_help.png
-rw-r--r-- 1 nelz users 3.2K Feb 28 10:57 Very_little_helps.tx
[nelz@hactar lxf/Answers 0]%
```

1 File permissions – this is a script as it has the execute bits set.

2 The user and group owning the file.

3 A directory usually has x set but also the special character d.

4 The time and date that the file was last modified.

5 Many distros add the `--color=auto` option, which helps distinguish between different types of file.

What goes where?

Users coming from Windows can be puzzled by the way Linux handles separate drives and partitions. Unlike the drive letter system used by Windows, Linux mounts everything in the same hierarchy. Your root partition, containing the core system files, is mounted at `/`, the root of the filesystem tree. Other partitions or drives can be mounted elsewhere at what are called mount points. For example, many distros use a separate partition for the home directory, where users' files are kept, to make installing a new version easier. This is a completely separate partition, it can even be on a different hard drive, but it appears at `/home` just as though it were part of the root partition. This makes everything easier and transparent for the user.

There is another difference. Linux, in common with every operating system but MS-DOS, uses a forward slash to separate directories. The layout of directories is also different, organising files according to their type and use. The main directories in a Linux filesystem are as follows...

`/` The root of the filesystem, which contains the most critical components.

`/bin` and `/usr/bin` General commands.

`/sbin` and `/usr/sbin` System administration commands for the root user.

`/etc` Where system configuration files are kept.

`/usr` Where most of the operating system lives. This is not for

user files, although it was in the dim and distant past of Unix and the name has stuck.

/lib and **/usr/lib** The home of system libraries.

/var Where system programs store their data. Web servers keep their pages in **/var/www** and log files live in **/var/log**.

/home Where users' data is kept. Each user has a home directory, generally at **/home/username**.

Moving around

Now that we know where everything is, let's take a look at the common commands used to navigate the filesystem. Before going anywhere, it helps to know where we are, which is what **pwd** does. Many Unix commands are short, often two to three characters; in this case, **pwd** is print working directory – it tells you where you are. Many distros set up the terminal prompt to display the current directory, so you may not need this command often. Moving around is done with the **cd** (change directory) command. Run it with no arguments to return to your home directory. Otherwise it takes one argument, the directory to change to.

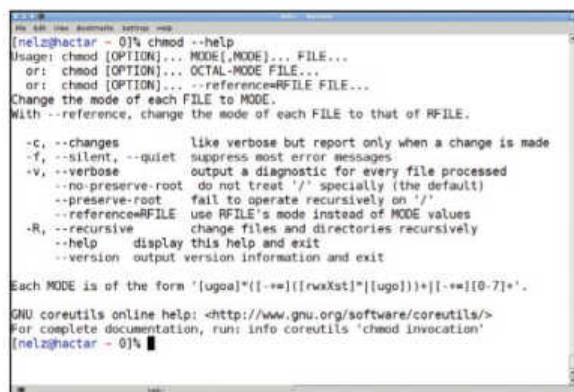
Directory paths can be either relative or absolute. An absolute path starts with **/** so **cd /usr/local** goes to the same place wherever you are starting from. A relative path starts at the current directory, so **cd Documents** goes to the Documents sub-directory of wherever you are, and gives an error if it is not there. That sounds less than useful if you can only descend into sub-directories, but there are a couple of special directory names you can use. To go up a directory use **cd ..** – a single dot is the current directory. There is also a shortcut for your home directory: **~**. Let's say you have directories called Photos and Music in your home directory and you are currently in Photos, either of these commands will move into Music:

```
cd ../Music
cd ~/Music
```

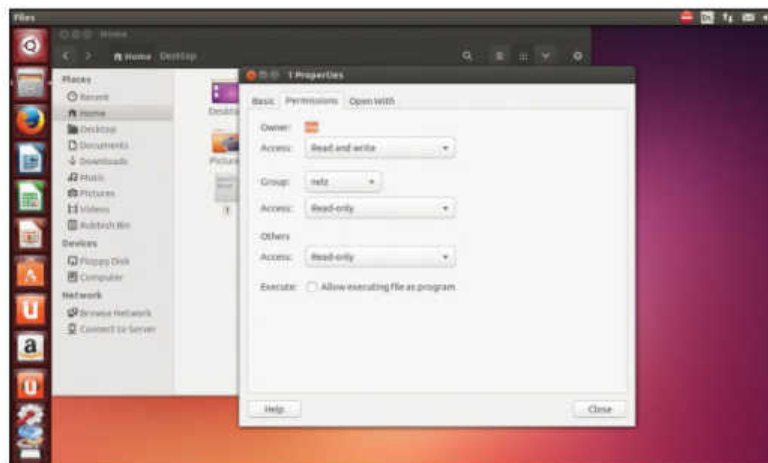
You can tell where you are with **pwd**, but how do you know what is in the current directory? With the **ls** command. Used on its own, it gives a list of files and directories in the current directory. Add a path and it lists the contents of that directory. If you want to know more about the files, use the **-l** (**--long**) option, which tells you the size and date of the file, along with information about ownership and permissions, which we will look at later.

With your permission

Every file object (that is files, directories and device nodes in **/dev**) has a set of permissions associated with it, as shown in



► If you need help with a command, ask the command for it. Most commands give a brief summary of their options when run with **--help**.



the screenshot of the output from **ls -l**. These are normally in the form **rw-rw-rw-r** and shown by **ls**, or the numeric equivalents. The three letters stand for read, write and execute, and are shown three times for the file's owner, the group it belongs to, and other users. For example, **rw-r--r--** is a common set of permissions for files; it means the owner of the file can read from or write to it, all other users can only read it. Program files usually appear as **rw-r-xr-x**, the same permissions as before but also all users can execute the file. If a program does not have execute permissions, you cannot run it. This is sometimes the case with system programs owned by the root user and only executable by root.

When applied to directories, the meanings are slightly different. Read means the same, but write refers to the ability to write into the directory, such as creating files. It also means that you can delete a file in a directory you have write permissions for, even if you don't have write permissions on the file – it is the directory you are modifying. You can't execute a directory, so that permission flag is re-purposed to allow you to access the contents of the directory, which is slightly different from read, which only allows you to list the contents (that is, read the directory).

File permissions are displayed by using the **-l** option with **ls** and modified with **chmod**, which can be used in a number of different ways, best shown by example:

```
chmod u+w somefile
chmod o-r somefile
chmod a+x somefile
chmod u=rw somefile
chmod u=rwx,go=rx somefile
chmod 755 somefile
```

The string following **chmod** has three parts: the targets, the operation and the permissions. So the first example adds write permission for the user. The next one removes read permission for other users, while the third adds execute permission for all users. **+** and **-** add and remove permissions to whatever was already set, while **=** sets the given permissions and removes the others, so the next example sets read and write for the file's owner and removes execute if it was previously set. The next command shows how we can combine several settings into one, setting read, write and execute for the owner, and read and execute for the group and others. The final command does exactly the same, but using the numerical settings. Each permission has a number: **4** is read, **2** is write, and **1** is execute. Add them together for each of the user types and you have a three-digit number that sets the permissions exactly (there is no equivalent to **+** or **-** with this method).

► Here is the GUI way of changing file permissions. You would need to do this for each file you wanted to change, and click a separate box for each permission.

get in action

```

Terminal
File Edit View Search Terminal Help

SUMMARY OF LESS COMMANDS

Commands marked with * may be preceded by a number, N.
Notes in parentheses indicate the behavior if N is given.
A key preceded by a caret indicates the Ctrl key; thus ^K is ctrl-K.

h H          Display this help.
q :q Q :Q ZZ  Exit.
-----

MOVING

e ^E j ^N CR * Forward one line (or N lines).
y ^Y k ^K ^P * Backward one line (or N lines).
f ^F ^V SPACE * Forward one window (or N lines).
b ^B ESC-v    * Backward one window (or N lines).
z          * Forward one window (and set window to N).
w          * Backward one window (and set window to N).
ESC-SPACE * Forward one window, but don't stop at end-of-file.
d ^D       * Forward one half-window (and set half-window to N).

HELP -- Press RETURN for more, or q when done

```

» **Less** displays text from any source – from a file, the output of another program or its built-in help if you manage to get stuck.

» **add-apt-repository** Adds extra repositories to the system.

» **dpkg** A lower level package manipulation command.

These commands generally require root (superuser) access, so should be run at the root user or with **sudo** – we will stick with the **sudo** approach here. We've already mentioned that repos are indexed, so the first thing to do is update your index files to match the current contents of the repositories with:

```
sudo apt-get update
```

Then you probably want to make sure that your system is up to date:

```
sudo apt-get upgrade
```

This will list the packages it wants to install, tell you how much space it needs for the download, and then get on with it when you tell it to. When you want to install some new software, unless you have been told the exact name to install, you may want to search for it first, like this:

```
apt-cache search gimp
```

This will spit out a long list of packages, because it searches both name and description, and lists anything mentioning gimp, and there are a lot of them. To search only the names, use the **-n** or **--names-only** option:

```
apt-cache search -n gimp
```

This often gives a more manageable output, but still a lot in this case, perhaps too much to fit in your terminal window. The solution to this is to pipe the output from this command to the program **less**:

```
apt-cache search -n gimp | less
```

The **less** command is a pager – it lets you read text page by page and scroll through it. It can be used with any program that generates lots of terminal output to make it easier to read (see the 'Package management' walkthrough opposite for more details). Once you have found the package you want, installation is as simple as:

```
sudo apt-get install gimp
```

You can install multiple programs by giving them all to **apt-get** at once:

```
sudo apt-get install program1 program2...
```

Not every program you try will be what you want, so you can tidy up your hard drive by uninstalling it with:

```
sudo apt-get remove program1
```

Or you can use:

```
sudo apt-get purge program1
```

Both commands remove the program, but **remove** leaves its configuration files in place while **purge** deletes those, too.

There are a number of extra options you can use with **apt-get**, the **man** page lists them all (type **man apt-get** in the terminal), but one of the most useful is **--dry-run**. This has **apt-get** show you what it would do without actually doing it, a useful chance to check that you are giving the right command. Remember, computers do what you tell them to, not what you want them to do! Finally, you don't normally need to use **dpkg**, but it is useful for listing everything you have installed with **dpkg -l**.

Terminal: Disk management

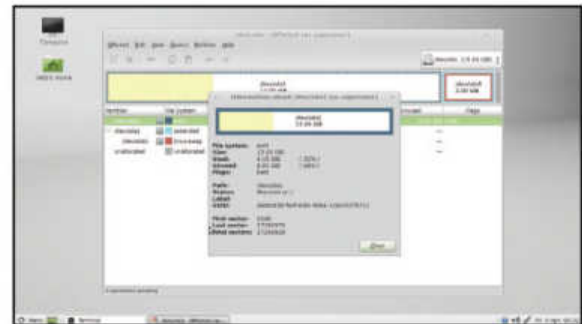
We continue our tutorials about using the terminal by explaining how to manage your disks, partitions and filesystems.

We store data in a variety of ways. On our hard disks, on optical discs (CDs and DVDs) and on removable devices such as USB sticks and external hard drives. In Windows, each would be given a drive letter, starting with C (A and B were reserved for floppy drives). You may see words such as partition and filesystem bandied about, so let's clarify them before we go any further. A disk is divided into partitions, separate physical areas each used to store data. Hard disks always have partitions, even when it appears you are using the whole disk, as in a Windows C drive, but there is a distinction. Removable devices such as USB sticks may or may not have partitions – most do. The only types of media that are never partitioned are floppy and optical discs.

Each partition contains one filesystem. A filesystem is a means of storing files, directories and their associated data – it organises the ones and zeros stored in the device's memory into useful objects. It is important to understand the differences between these terms before we continue, even though they are often used interchangeably.

Types of filesystem

There are several different filesystems supported on Linux. A native filesystem is one that is designed for use with Linux or other Unix-like systems. The most commonplace of these is ext4 (along with its predecessors ext3 and ext2 that are still in



There are graphical tools for managing partitions and filesystems, the most popular of which is **GParted**.

use). There are other native filesystems such as reiserfs, BTRFS and XFS. Why do we have these variants? Because not every system has the same needs; a mail server needs to store many thousands of small files, a database server works with a smaller number of much larger files, and requires access to random points in those files. A laptop's filesystem needs to be resilient to unplanned shutdowns, such as when a battery fails. So we end up with different filesystems for different purposes, although for general use ext4 does a fine job. In fact, the enhancements to ext4 over ext3 mean it works well in a wider variety of situations and is a sensible choice for a standard desktop or laptop system.

Creating partitions and filesystems

```
[neil@hacker: Terminal/Tutorials/Part3 8]% sudo mkfs.ext4 /dev/sdb1
mke2fs 1.42.9 (28-Dec-2013)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
3276800 inodes, 13107200 blocks
655360 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=4294967296
400 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 894736, 1605632, 2654208,
    4096000, 7962624, 11239424

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

[neil@hacker: Terminal/Tutorials/Part3 8]%
```

Here we're creating an ext4 filesystem using the standard defaults. Get the partition right, because **mkfs** doesn't ask whether you are sure.

Partitioning a drive is normally done by a distro's installer, but there are graphical and command-line programs to work with partitions and filesystems. At the command line, **fdisk** and **gdisk** are the standard commands, the former working with old style MS-DOS partition tables, while **gdisk** works with the newer, more reliable and flexible GPT partitions. They work in much the same way. See the step by step guide for more information.

Once you have created partitions, you need to make filesystems on them, which is done with the **mkfs** range of commands. These all have names of the form **mkfs.FSTYPE**, such as:

sudo mkfs.ext4 /dev/sdb1

There are different options you can use for each filesystem type, depending on that particular filesystem's features, so you will have to consult the **man** page for that flavour of **mkfs** to see what you need, although they all have sensible defaults. There is also a generic **mkfs** command that takes a **-t** option to specify the filesystem type, but the versions with the filesystem added save having to remember the exact name. The main ones you will need are **mkfs.ext4**, **mkfs.ext3** and **mkfs.vfat**, with the last one being used for OS-independent media, such as flash drives.

Step-by-step: Partition a disk



1 sudo gdisk /dev/sda

You need to use **fdisk** for hard drives partitioned with the old MBR system (**gdisk** will warn you if this is needed). You must give the name of the drive, then press [m] in **fdisk** or [?] in **gdisk** for a list of all the main commands. Any changes you make here stay in memory and are only written to the disk when you press [w].

Non-native filesystems are those developed for use with other platforms, but supported in Linux. The most notable of these is FAT, the default Windows filesystem for many years and still the standard choice for USB sticks, camera memory cards and other removable media. Also supported to a certain extent are the newer exFAT and NTFS filesystems from Microsoft and Apple's HFS and HFS+ filesystems. The collection of supported filesystems is rounded out by the likes of ISO9660 and UDF, used by CD and DVD media.

Most of the time, you don't need to worry about the type of filesystem used on a media device, just plug it in and the Linux kernel recognises the filesystem in use.

Mounting a filesystem

Making the contents of a filesystem available to the OS and user is called mounting. In Linux, each filesystem is mounted at a particular point within the filesystem hierarchy, known as the mount point. The root filesystem, the one containing the core OS components, is mounted at **/**, the root of the filesystem tree. Many distros use a separate partition and filesystem for the user's home directories; it separates the OS from the user's files so you can update, re-install or switch the OS without affecting your personal files. Users' home directories reside at **/home/username**, so a separate home filesystem would be mounted at **/home**, the mount point for the home filesystem. This directory must exist on the root filesystem but is usually empty – as soon as the home filesystem is mounted, its contents appear at **/home**. Anything that was there before is no longer visible, but is still there and will reappear when home is unmounted.

There are three main ways to mount a filesystem: at boot, manually or automatically. The system filesystems, such as **/** and **/home**, are mounted automatically at boot, using information stored in the filesystem table **/etc/fstab** – remember, **/etc** is where system settings are stored. Here is a typical **fstab** entry for mounting the **a** partition.

```
/dev/sda2 /home ext4 defaults 0 2
```

Each filesystem's entry is on a single line, with six fields:

1. Device name Linux disks are named **sda**, **sdb** and so on, with the partition's number added to the disk's device, so **/dev/sda2** is the second partition on the first disk.

2 Create some partitions

Press [n] to create a partition – you normally can press [Enter] to accept the defaults for partition number and start point. The default end point is the end of the available space – if you want to use a specific size, enter **+** followed by the size you want, for example **+50G**. For anything but a Linux partition, you should give its type; entering **L** lists them.

3 Show the partition layout

Press [p] to list the current partition layout, either the existing layout from a freshly loaded disk or the layout you have created in **fdisk**/**gdisk**. Make sure everything is as you want before you use [w] to commit to the disk, as there is no going back from [w]. Until then, you can make further changes or press [q] to abort without writing anything to disk.

2. Mount point Where the filesystem is to be mounted – it must exist.

3. Filesystem type It is possible to use **auto** here and let the kernel figure it out for you, but with a fixed disk it makes sense to be specific.

4. Mount options The keyword **defaults** means use whatever is standard for that filesystem. Specific settings can be given here, separated by commas.

5. Dump Used for a particular type of backup, normally left at zero.

6. This is used by **fsck** to set the order in which filesystems are checked when booting. Set to **1** for the root partition and either **2** or **0** for others – **0** means do not check.

Mounting manually is done with the mount command:

```
sudo mount /dev/sda4 /mnt/backup -t ext4
```

The first two options are the device and mount point, **-t** specifies the filesystem type; if you omit this, **auto** is used. You can also use **-o** to add other options, such as:

```
sudo mount /dev/sda4 /mnt/backup -t ext4 -o noatime
```

The **noatime** option is often used in **fstab** to improve performance; it reduces the number of writes to the disk by only recording when files are modified, not simply read. If a device is listed in **fstab**, you need only give either the device or mount point, and **mount** will take the rest from **fstab**:

```
sudo mount /dev/sda3
```

or

```
sudo mount /mnt/music
```

We have used **sudo** in all of these examples because mounting is generally only permitted by the root user. This behaviour can be changed for mounts listed in **fstab** by adding **user** or **users** to the options:

```
/dev/sda3 /mnt/music ext4 users,noatime 0 0
```

Now any user can mount this with:

```
mount /mnt/music
```

The difference between **user** and **users** is that with the former, only the user who mounted the filesystem can unmount it. Speaking of unmounting, that is done with the **umount** command, followed by either the mount point or device. This command also accepts multiple paths to unmount several at once:

```
sudo umount /mnt/music /mnt/photos /dev/sda5
```


Compression types

There are two types of compression: lossless and lossy. Lossy compression achieves a greater reduction in file sizes by carefully discarding data that doesn't greatly affect the results. JPEG and MP3 files, for example, use lossy compression. Archiving uses lossless compression and uncompresses the result to get you back exactly where you started. There are a number of lossless compression methods available. Here are some of the most common.

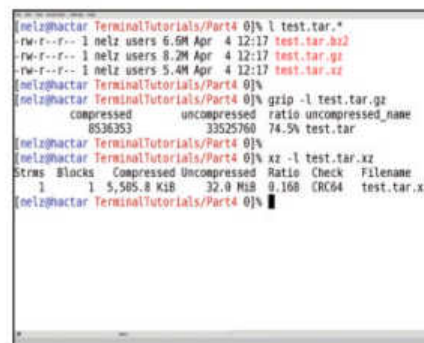
» **Deflate** This is the default compression used by *Zip*; it is old and not particularly effective, but is fast and well supported.

» **Compress** This is an old Unix compression program. Its files have a .Z extension, but you are unlikely to see many in the wild. There are also patent issues with it.

» **Gzip** This *Compress* replacement is completely open and still in use today. *Gzip* uses Deflate, but generally produces smaller archives than *Zip* with its default settings. It does not give the best compression, but it compresses and decompresses quickly, making it a good choice when size is not the most vital consideration.

» **Bzip2** This is a more efficient compressor, but at the expense of speed. Compression in particular can be slower, but it produces more compact results.

» **xz** This more recent program uses the LZMA2 compression algorithm also used by 7-Zip. It is fast – particularly for decompression – and works brilliantly. It is the compressor of choice for many key Linux projects, including Coreutils and the kernel itself



Here is the same archive using different compression programs. The *Gzip* version is significantly larger, but faster to compress and decompress.

which particular type of compression to use: **z** uses gzip compression, **j** uses bzip2 compression and **J** uses xz compression. (Watch the capitalisation!)

There are also long versions of these arguments that make the commands more readable, but most of us are lazy and use the version that is shorter to type. However, we could also have used this command line if we wanted to:

```
tar --create --gzip --file foo.tar.gz foo
```

The file extension is not required, but it's a convention that makes it easier for people to see what type of archive it is – the system itself needs no such help as it can work all this out for itself. Unpacking an archive is simply a matter of replacing **c** with **x**, or **--create** with **--extract**. However, you don't need to give the compression type, as *Tar* figures it out:

```
tar xf foo.tar.gz
```

Another option you may want to add is **v** or **--verbose**, to show you what *Tar* is doing. If you have been given a tarball, you may want to see what is inside it without unpacking it. If you have created an archive, particularly a backup, you may want to check it's correct before relying on it. The test option checks the integrity and lists the contents of the archive.

```
tar tvf foo.tar.gz
```

Those are the main *Tar* options, but the program has many more, such as **A** or **--concatenate** to add files to an existing archive instead of creating a new one.

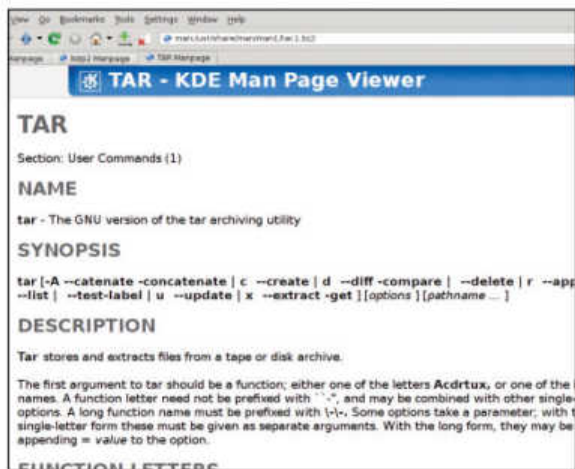
Future proofing

We mentioned that *Tar* can handle any new compression format that comes along because it passes compression to another program. There are command line options to do this automatically for *gzip*, *bzip2* and *xz*, but what if someone comes up with a new compressor? Say something like **sdcc** – super-duper compressor? You could create an uncompressed tarball and then use **sdcc** to compress it, but that’s wasteful and slow, so instead use a pipe:

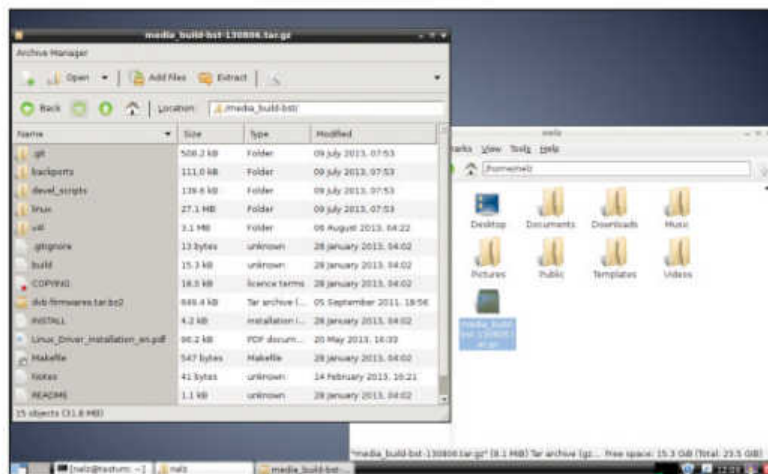
```
tar c foo | sdc >foo.tar.sdc
```

```
unsdc foo.tar.sdc | tar xv
```

Here, we use only the **--create** option with *Tar*. The lack of destination causes *Tar* to send the archive data to standard output, which is then piped to the **sdc** compressor program. The second command reverses the process, decompressing the archive and sending it to *Tar* for extraction.



► **Tar and the compressors have man pages, with many options. You'll usually only need the ones covered here.**



Most environments can view the archive contents. Shown above is the result of double-clicking a tarball in LXDE (the preferred desktop for Raspberry Pi).

Terminal: Core programs

Out of the hundreds of different terminal commands available, here's a handy summary of some of the more useful ones.

We've looked at various shell commands in the last few tutorials, but they have each been in the context of performing a particular task. It's time to take an overview of some of the general-purpose commands. There are thousands of terminal commands, from the commonplace to the arcane, but you need only a handful of key commands to get started. Here we will look at some of the core workhorse commands, giving a brief description of what each one is for. As always, the **man** pages give far more detail on how to use them. Many of these produce more output than can fit in your terminal display, so consider piping them through **less**.

Central to any terminal activity is working with files, creating, removing, listing and otherwise examining them. Here are the main commands for this.

» **ls** Lists the contents of the current or given directory.

» **ls -l** As for **ls**, but gives more information about each item. Add human-readable file sizes with **-h**:

```
ls -lh MyPhotos
```

» **rm** Deletes a file. Use the **-i** option for confirmation before each removal or **-f** to blitz everything. With **-r** it deletes directories and their contents, too.

» **rmdir** Deletes a directory, which must be empty.

» **df** Shows free disk space on all filesystems or just those given on the command line.

» **du** Shows the amount of space used by individual files or directories.

```
df -h /home
```

```
du -sh /home/user/*
```

» **file** Identifies the type of a file. Unlike Windows, which uses the file name extension, this command looks inside the file to

see what it really contains.

» **find** Searches the current or given directory for files matching certain criteria. For example, you could find all *LibreOffice* spreadsheets with:

```
find Documents -name '*.ods'
```

» **locate** This also looks for files but using a much faster system. The **locate** database is automatically rebuilt each day by *updatedb*, and **locate** then searches this. It's fast, but doesn't know about very recent changes.

Text handling

Text files are all around us, from emails to configuration files, and there are plenty of commands that deal with them. If you want to edit a text file, for example, there are a number of choices, with the two big ones being **Emacs** and **Vi**. Both are overkill if you just want to tweak a configuration file; in this instance, try **nano** instead:

```
nano -w somefile.txt
```

The **-w** option turns off word wrapping, which you certainly don't want when editing configuration files. The status bar at the bottom shows the main commands – for example, press [Ctrl]+[X] to save your file and exit.

This assumes you know which file you want, but what if you know what you're looking for but not the name of the file? In that case, use **grep**. This searches text files for a string or regular expression.

```
grep sometext *.txt
```

will search all .txt files in the current directory and show any lines containing the matching text from each file, along with the name of the file. You can even search an entire directory hierarchy with **-r** (or **--recursive**):

Getting help

The command line may appear a little unfriendly, but there's plenty of help if you know where to look. Most commands have a **--help** option that tells you what the options are. The **man** and **info** pages are the main sources of information about anything. To learn all the options for a program and what they do, run:

```
man progname
```

The **man** pages are divided into numbered sections. The ones that are most applicable to using the system are:

» **1** User commands

» **5** File formats and conventions

» **8** System administration tools

If you don't specify the number, **man** will pick the first available, which usually works. But **man** pages are not limited to programs; they also cover configuration files. As an example, passwords are managed by the **passwd** command, and information is stored in the **/etc/passwd** file, so you could use:

```
man passwd
```

```
man 1 passwd
```

```
man 5 passwd
```

The first two would tell you about the **passwd** command, while the third would explain the content and format of the **/etc/passwd** file.

Man pages have all the information on a single page but **info** pages are a collection of hypertext-linked pages contained in a single file. They often provide more detail but aren't very intuitive to read – try **info info** to see how to use them. It's often easier to use a search engine to find the online version of **info** pages, which contain the same information in the more familiar HTML format.

Commands, options and arguments

You'll often see references to command arguments and options, but what are they? Options and arguments are the things that tell a program what to do. Put simply, arguments tell a command what to do, while options tell it how to do it – although the lines can get a little blurred. Take the **ls** command as an example – this lists the contents of a directory. With no options or arguments, it lists the current directory using the standard format:

```
ls
Desktop Downloads Music Public
Videos
Documents examples.desktop Pictures
Templates
```

If you want to list a different directory, give that as an argument:

```
ls Pictures
```

or

```
ls Desktop Downloads
```

Arguments are given as just the names you want listed, but options are marked as such by starting with a dash. The standard convention among GNU programs, and used by most others, is to have long and short options. A short option is a single dash and one letter, such as **ls -l**, which tells **ls** to list in its long format, giving

more detail about each file. The long options are two dashes followed by a word, such as **ls --reverse**, which lists entries in reverse order, as is pretty apparent from the name. **ls -r** does the same thing but it is not so obvious what it does. Many options, like this one, have long and short versions, but there are only 26 letters in the alphabet, so less popular options are often available only in the long version. The short options are easier to type but the long ones are more understandable. Compare

```
ls -l --reverse --time
```

with

```
ls -l -r -t
```

or even

```
ls -lrt
```

Each gives a long listing in reverse time/date order. Notice how multiple short options can be combined with a single dash. While we're talking

» By piping the output from **du** through **sort**, and adding extra options to both commands, we can see which directories use the most space.

about **ls**, this is a good time to mention so-called hidden files. In Linux, any files or directories beginning with a dot are considered hidden and do not show up in the output from **ls** or in most file managers by default. These are usually configuration files that would only clutter up your display – but if you want to see them, simply add the **-A** option to **ls**.

```
grep -r -I sometext somedir
```

Be careful when you are searching large directory trees, because it can be slow and return strange results from any non-text files it searches. The **-I** option tells **grep** to skip such binary files.

Text is also the preferred way of passing data between many programs, using the pipes we looked at previously. Sometimes you want to pass data straight from one program to the next, but other times you want to modify it first. You could send the text to a file, edit it and then send the new file to the next program, or you could pass it through a pipe and modify it on-the-fly. **Nano** edits files interactively, **grep** searches them automatically, so we just need a program to edit automatically; it's called **sed** (Stream EDitor). **Sed** takes

a stream of text, from a file or pipe, and makes the changes you tell it to. The most common uses are deletion and substitution. Normally, **sed** sends its output to **stdout**, but the **-i** option modifies files in place:

```
sed -i 's/oldtext/newtext/g' somefile.txt
```

```
sed -i '/oldtext/d' somefile.txt
```

The second example deletes all lines containing **oldtext**.

Another useful program is **awk**, which can be used to print specific items from a text file or stream.

```
awk '{print $1}' somefile.txt
```

```
cat *.txt | awk '/^Hello/ {print $2}'
```

The first example prints the first word from each line of the file. The second takes the contents of all files ending in **.txt**, filters the lines starting with **Hello** (the string between the slashes is a pattern to match) and then prints the second word from each matching line.

Networking

We normally think of big graphical programs like *Chromium* and *Thunderbird* when we think of networked software, but there are many command line programs for setting up, testing and using your network or internet connection.

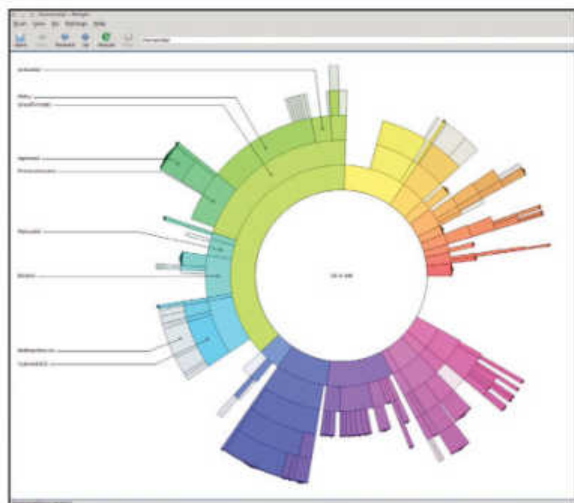
» **Ping** Sends a small packet to a remote server and times the response, which is useful if you want to check whether a site is available, or if your network is working.

```
ping -c 5 www.google.com
```

» **wget** Downloads files. The only argument it needs is a URL, although it has a huge range of options that you will not normally need.

» **hostname** Shows your computer's host name, or its IP address with **-i**.

» **lynx** A text mode web browser. While not as intuitive as *Chromium* or *Firefox*, it is worth knowing about in case you ever suffer graphics problems.



» KDE's **Filelight**, shown here, and Gnome's **GDU** provide a graphical alternative to **du**, representing disk usage visually.

Terminal: Time-savers

The terminal has a host of handy tricks that will save you time and make your life easier. Let's introduce a few of the most useful techniques.

Earlier, we mentioned that using the terminal can be faster and more productive than using a GUI, especially with repeated operations. To get the most out of this, it helps to know a few of the shell's secrets.

Do you often find yourself typing in the same command options over and over again? Do you sometimes have trouble remembering the correct options? Are you just plain lazy and want to avoid typing wherever possible? If you answered yes to any of these, aliases are for you. You already have some shell aliases set up. To list them, type:

```
alias
```

which will return entries like this (from Ubuntu):

```
alias ll='ls -alF'
```

```
alias ls='ls --color=auto'
```

The first of these examples is simple – if you type in **ll**, it runs **ls -A**. The second is cleverer, because the alias is the same as the command name, so the old **ls** behaviour disappears and it's always run with the **--color=auto** option. If you find yourself always using certain options with particular commands, aliases like this effectively make those options the defaults. The alias is expanded before the rest of the command line is interpreted, so:

```
ll X Y Z
```

becomes

```
ls -alF X Y Z
```

These aliases aren't always easy to remember, but fear not – you can create your own aliases using exactly the same syntax as in the output from the **alias** command:

```
alias myalias='somecommand --option1 --option2'
```

```
nelz@shooty ~/lxf/TerminalTutorials $ alias
alias cp='cp -ip'
alias df='df --human-readable --no-sync --print-type'
alias du='du -sch'
alias egrep='egrep --colour=auto'
alias fgrep='fgrep --colour=auto'
alias grep='grep --colour=auto'
alias jc='sudo journalctl'
alias l='LC_TIME="POSIX" ls -lha --color=auto'
alias lmount='sudo mount -o loop'
alias ls='ls --color=auto'
alias mm='make menuconfig'
alias msg='sudo tail -f /var/log/messages'
alias mv='mv -i'
alias poff='sudo /usr/sbin/poff'
alias pon='sudo /usr/sbin/pon; sleep 10; sudo /usr/sbin/plog'
alias pss='pgrep --full --list-full'
alias rm='rm -i'
alias sc='sudo systemctl'
alias vmerge='sudo emerge --update --deep --changed-use --ask --with-bdeps y --keep-going @system @world'
nelz@shooty ~/lxf/TerminalTutorials $
```

» Aliases save you typing and enable you to use more memorable command abbreviations. You choose these yourself, so you have no excuse for forgetting!

Aliasing a command to its own name is a neat trick, but what if you then need to use the original command? Don't worry – the developers have thought of that. Prefix the command with a backslash and your alias will be ignored:

```
\ls Documents
```

Profiles

You can customise your terminal experience with aliases and custom prompts, but to make them even more convenient, you need a way of applying these automatically when you open a terminal. That can be done through your profile, which is a file containing commands that are read and run whenever you open a shell session.

There are several locations that are read, the first of which is **/etc/profile**, which contains global profile settings. This in turn runs any files in **/etc/profile.d**, which makes it easy to add global settings without touching the default profile. Then the user's profile is read from one of **~/.bash_profile**, **~/.bash_login** and **~/.profile**. Only the first of these files that exists is run, and any settings in here override those in the global profile if the same thing is set in both.

The profile is simply a set of shell commands, one per line, that are run when the shell starts up. These can set up aliases or environment variables, or set the command prompt. A typical use of environment variables would be to change the default text editor to **nano**:

```
EDITOR=/usr/bin/nano
```

Word completion

If you aren't a particularly confident touch typist, you'll soon get fed up with typing out command and file names in full, then dealing with the error messages resulting from misspellings. Fortunately, the shell provides a way to both save typing and avoid mistakes, called tab completion. The name is fairly self-explanatory – it uses the [Tab] key to complete the word you are typing. If this is the first word on a line, it will be a command, so it looks in the command path for matches – for example, typing **chm** and hitting [Tab] will complete to **chmod**. Used after the command, hitting [Tab] completes file names, so instead of typing:

```
cat /path/to/somelongdirectory/someevenlongerfilename.txt
```

you can use:

```
cat /pa[Tab]som[Tab]som[Tab]
```

Isn't that easier? When more than one file matches, the [Tab] key will complete up to the point where it becomes ambiguous; pressing [Tab] again at this point shows a list of options, which you can cycle through with [Tab], or add

another letter or two and press [Tab] again. It is actually a lot easier to use than to describe – just try it.

Wild, wild cards

There are times you want to include more than one file in a command, but don't want to type them all out. The Linux shell has a couple of wildcard operators to help you out with this. The most frequently used one is the star, which is used as a replacement for any string of characters. So ***** on its own will match every file in the current directory (note that this is different from MS-DOS shells, where ***** does not match **.** and you need to use ***.*** to match any file with an extension). You can also use ***** to match one part of a file name – ***.txt** matches all files that end with **.txt**, and **a*.txt** only matches those that also start with **a**. This matching is performed by the shell. It expands the matches to a list of files before passing them to the command, which never sees the wildcard or even knows that you've used one. This can be an important distinction, especially if you want to pass a ***** to the program you're running. For example, if you're using **scp** to copy files from a remote computer and run:

```
scp user@othercomputer:Documents/*.txt Documents/
```

you may expect it to copy all **.txt** files from the other computer's Documents directory to the same one here, but it won't. The shell expands the ***.txt** to match all **.txt** files on the local system, so you only copy the files you already have. In such an instance, you need to tell the shell not to expand the ***** by preceding it by a backslash, known as escaping it:

```
scp user@othercomputer:Documents/*.txt Documents/
```

Now it passes **Documents/*.txt** unchanged to the remote computer and lets its shell handle the expansion.

There's a second wildcard character. While ***** matches any number of characters, including none, **?** matches exactly one. So **ab?.txt** matches **abc.txt** and **ab1.txt** but not **ab.txt** or **abcd.txt**, whereas **ab*.txt** would match all of them.

Bring on the substitute

There is a powerful feature in shells called command substitution. This allows you to embed the output from one command in another. As a simple example, there is a command called **foo** on your system, and let's say you want to know what type of program it is: compiled, Python or even a shell script. The **which** command gives you the path to a program, and **file** reports the type of a file. You could use:

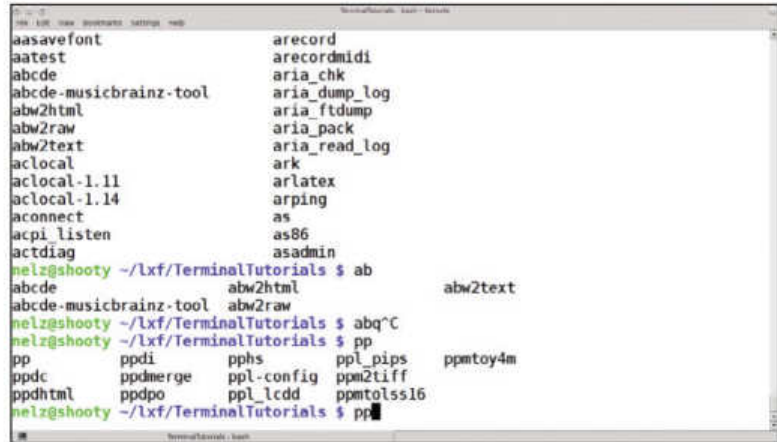
```
which foo
to get /usr/local/bin/foo and then
file /usr/local/bin/foo
to get the file type, but even with copy and paste, this is a lot of work and easy to mistype. Alternatively, you could do:
file $(which foo)
/usr/local/bin/foo: a /usr/bin/perl script, ASCII text
executable
```

What happens is that the shell runs whatever is inside **\$(..)** and substitutes the output from that into the command line before running it. You can also use backticks for this:

```
file `which foo`
```

This is quicker to type but less readable, especially when posted to a forum where the reader's font can make **`** and **`** look similar. With the first format there is no such confusion.

One of the great advantages of the shell is that it's simple to repeat commands, no matter how complex, with a couple of keypresses. At its simplest, you can press the [Up] key to show previous commands, then press [Enter] to run your chosen one again. You can also edit the command before running it, if you like. That's fine if you want to use one of the



As the name suggests, **tab completion** lets you use the [Tab] key to complete the word you're typing, helping save time and avoid mistakes.

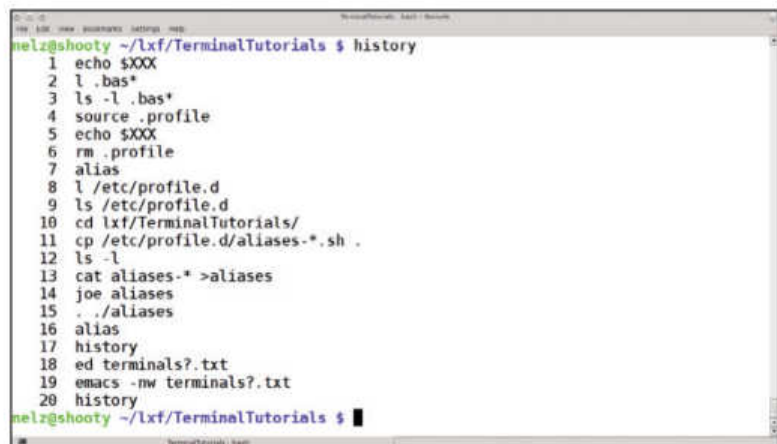
last few commands you ran, but most shells keep a history of the last 500 or so commands that you can access again.

Please repeat that

To avoid wearing out your keyboard's [Up] arrow key (and your eyes), press [Ctrl]+[R] and then type in part of the command. The terminal will now show you the most recent command that matches what you are typing, updating with each keypress you make. If you have run several very similar commands, you don't need to keep typing – just type a few characters and press [Ctrl]+[R] again to see the previous match.

Keep going until you find the one you want. (Don't be alarmed – this is one of those things that takes far longer to describe than to actually do.) You can edit the command before running it. The search term that you type in doesn't have to be the command itself – any part of the command line will match, so you could start with the file name.

You can also access commands within the history directly. The **!** character tells the shell that you are referring to its history. **!!** is a shortcut for the previous command, while **!-n** means 'execute the command run n commands ago' (so **!!** is the same as **!-1**). **!xyz** runs the last command starting with **xyz**, and **!?xyz** does the same for the last command containing **xyz** (much the same as [Ctrl]+[R]). ■



The **history** command shows recently executed commands – use **!** and [Ctrl]+[R] to find and run again the commands you used before.

Terminal: User accounts

Whether you want to have one user or a hundred on your computer, it's time to explain the basics of user accounts on Linux.

Linux is a multi-user operating system, even if you are the only person using your computer. The most basic of systems has two users: you and the superuser, also called root. Every file or directory is owned by a user and has settings, called permissions, that specify who can read or write to it. This safeguards your files from being overwritten by another user, or possibly even read by them if so set. It also safeguards system files because they are owned by root and can only be changed by root. This includes writing to system directories, so only the root user can install new software there. So how do you install software? See the 'Becoming root' box opposite for the answer.

One word of caution when using data on multiple computers, such as with an external hard disk. While you may see your username as a name, say johnny99, the computer sees and stores it as a number, a UID or user ID. During installation, your distro will have created a root user, who always has a UID of 0, and a normal user. Most distros start at 1000 for the first general user, but some start at 500. The point is, it's that number stored on the disk as the owner of a file, so the same username may not own the file when you move the disk to another computer.

Create a user

Every user has a home directory. This is usually **/home/username** but it can actually be anywhere – the user created

to run a web server will have a home directory somewhere like **/var/www**. In addition to users, Linux also has groups. A group is basically a collection of users. For example if you have a USB scanner on your computer, you often need to be a member of the scanner group to be able to use it. Now that we understand usernames, groups, UIDs and home directories, we can create a user:

```
sudo useradd -m -c "John Smith" -g users -G scanner,audio john
```

We use **sudo** here because only the root user can create users. The **-m** option creates a home directory at **/home/john**, **-c** specifies a comment to store for the user, which is usually the user's full name, **-g** sets the primary group of the user, while **-G** adds secondary groups. Finally, we give the username. Not all of these options are necessary – if you omit **-g**, a default group will be used. Some distros use a single group called **users** for all non-system users, while others create a separate group for each user. The **groupadd** command works in a similar way, as do both of their counterparts – **userdel** and **groupdel**.

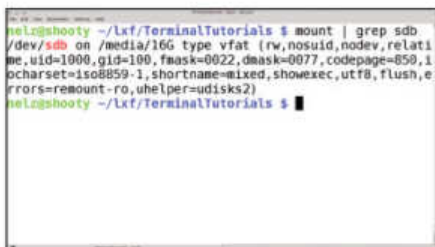
Add a password

We have created a user but he cannot log in yet because we haven't given him a password.

```
sudo passwd john
```

will ask you for the password twice. The **passwd** command

Different filesystems



▶ Running the **mount** command on its own shows the options a filesystem was mounted with. Here you see the **mask** setting for a FAT formatted USB stick (**fmask** and **dmask** do the same as **umask**, but only for files and directories respectively).

One area that causes confusion is that of permissions of mount points. It makes no difference what permissions and ownership are set on a mount point before a filesystem is mounted there; it acquires the permissions of the root directory of the filesystem you mount there. How you change things for the top-level directory of a filesystem depends on its type. In the case of Linux filesystems, this is simple: mount it and then use **chmod** or **chown** to set it up as you want. These settings will then apply whenever, and wherever, you subsequently mount that filesystem.

Windows filesystems, such as FAT on USB sticks, are treated differently. FAT has no concept of file ownership and NTFS has a

system which is incompatible with Linux permissions, so the filesystem driver imposes default permissions. These generally set all files and directories to be owned by the user who mounted the device, otherwise you couldn't write to it. File permissions usually default to **rw-r-xr-x** (or **755** if you prefer) by the automounter.

If you are mounting a filesystem manually, use the **uid** option to set ownership and **umask** for default permissions, like this:

```
sudo mount /dev/sdb1 /mnt/removable -o uid=john,umask=022.
```

The **umask** is subtracted from **777** to give the permissions – **755** in this case. A **umask** of **0** enables all permissions.

Becoming root

Almost all of the user management commands require root access. The superuser, often referred to as root, can do anything anywhere, regardless of any permissions. So how does a humble user do anything that requires root privileges? There are two ways of doing this.

The traditional way is to use the **su** (for switch user) command, which allows you to become another user, provided you know their password. Run **su** in a terminal (it switches to root if you don't specify a user), and it will ask for the root password. Once you have entered this, you are root in that terminal session, and can do anything you like – muhahaha!

Of course, this is a bit risky, especially if you don't log straight out after doing what you need. It also means that you have to give the root password, and therefore full access, to any user who needs to run a root command – so **sudo** was invented. **Sudo** is used to run individual

commands as root (or another user) but does not require you to know the root password. Instead, the root user must have set things up to allow you to run some or all commands as root, so it asks for your password. The normal installation of most distros allows the first user created to have **sudo** rights to everything, so you can install software, or do anything else, without ever having to log in as root, or spread the root password. In fact, some distros have no root password by default, and **sudo** is the only way to run administrative commands.

Sudo can be used to give individual users the right to execute only specific commands, plus it records every command it runs in the system log, along with who ran it, giving a far more secure and accountable system.



► With **sudo**, each user can have all, some or no administrative rights. This is a typical definition that allows members of the admin group to do anything.

can also be used to change the password of an existing account. If you run it without **sudo** or a username, however, it will change your own user's password, because only root can set passwords for anyone else. It is considered good practice to change passwords regularly, and you can enforce this by using **passwd**:

```
sudo passwd --maxdays 60 -warndays 7 john
```

This password will become invalid after 60 days and john will be warned about the expiry a week before. The usual rules about passwords – make them long and mixed case, preferably with numbers – apply here, and doubly so to the root password, the key to the kingdom.

User details are stored in **/etc/passwd**, which may be edited should you wish to change them. However, making a mistake could prevent you from logging in, so use

```
vipw
```

to edit it. This loads a copy of **/etc/passwd** into your preferred editor (as defined in **\$EDITOR**) and checks its

validity when you save it, before replacing the existing file. The format of **/etc/passwd** is explained fully with:

```
man 5 passwd
```

Transfer ownership

If you want to change the owner of a file, you need **chown**:

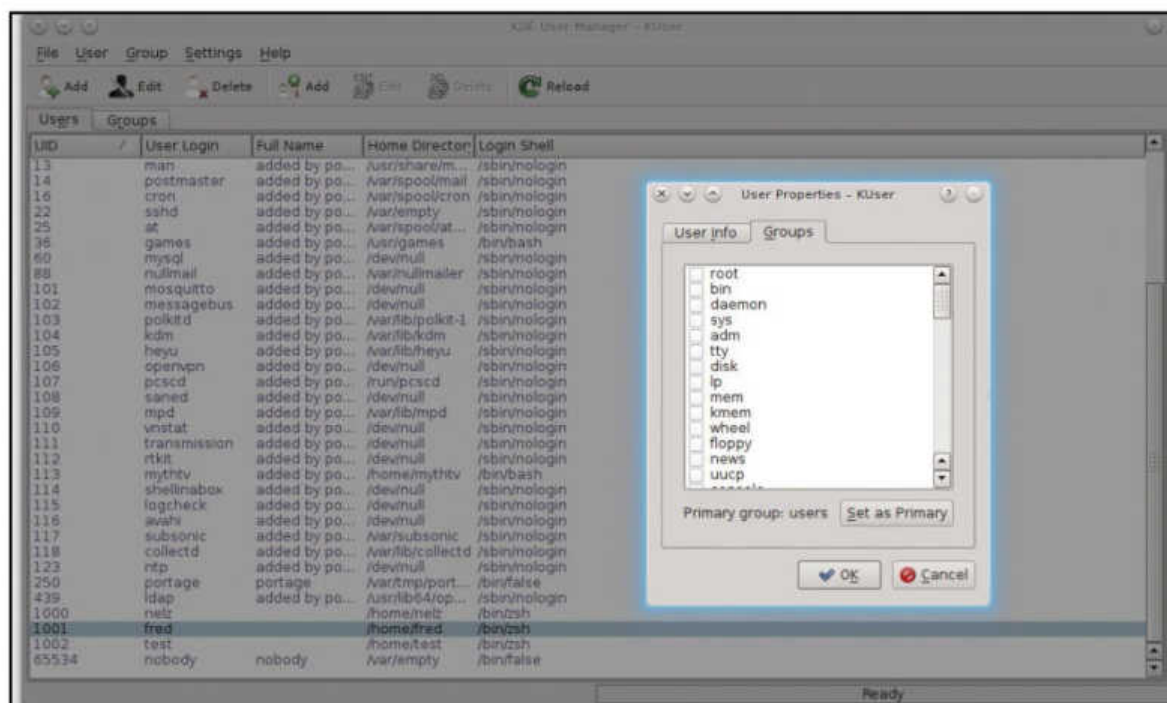
```
chown john somefile
```

```
chown john:users someotherfile
```

```
chown john: someotherfile
```

```
chown -R john: somedir
```

The first makes john the owner of a single file. The second command also changes the group. If you do not give a group after the colon, as in the third example, the group is changed to the user's default group. When applied to a directory, the **-R** option also changes all files and sub-directories in that directory. You can change just the group with **chgrp**. These commands must be run as root. Changing file permissions is done with **chmod**.



► Most desktops provide graphical alternatives for user management – this is KDE's Kuser – but they still need root privileges.

Man pages: Accessing help

Need more advice? Then you need to browse the ultimate collection of useful self-help books that resides inside the Linux operating system.

RTFM has long been considered the battle cry of the supposed Linux experts, and is claimed to scare new users away. If you haven't come across it before, it stands for something like 'Read the fine manual'. It is perhaps easy to appreciate the frustration some individuals feel when asked a question for the umpteenth time when the information is clearly covered in the manual. However, it's only possible for a user to read a program's documentation if they know where to find it. Happily, there are some sources of help within Linux, so let's look at each of them.

Before you even look for the manual, remember that many programs have built-in help. Run the command from a

terminal with the **--help** option to see a synopsis of its options. This applies to GUI programs, as well as shell commands. For example:

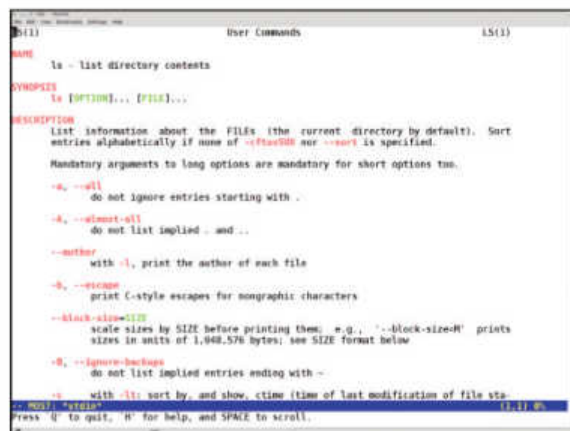
```
firefox --help
```

If you need more detail, then it may be time to RTFM, which on Linux systems usually means the man page. Man pages document just about everything on your system and are viewed by keying in the **man** command. If you want to know how **man** itself works, the classic recursive example is to open a terminal and run:

```
man man
```

A man page is a single page containing the reference documentation for its topic. The **man** command renders this document as readable text and displays it in your default pager, usually *less*. This means you use the same keyboard controls as *less* for navigating the page: cursor keys to scroll up and down, [Spacebar] to page down, and so on. Some man pages can be very long, so try **man bash** to enable you to search. In *less*, press [/] to start searching (or [?] if you want to search backwards), followed by your search term. Then use [n] to jump to the next match or [N] for the previous one. The man pages are divided into sections:

- 1 User commands
- 2 System calls
- 3 C library functions
- 4 Devices and special files
- 5 File formats and conventions
- 6 Games *et al*
- 7 Miscellany
- 8 System administration tools and daemons



```
ls(1)
NAME
  ls - list directory contents
SYNOPSIS
  ls [OPTION]... [FILE]...
DESCRIPTION
  list information about the FILEs (the current directory by default). Sort
  entries alphabetically if none of -ftaoSdu nor --sort is specified.
  Mandatory arguments to long options are mandatory for short options too.
  -a, --all
    do not ignore entries starting with .
  -A, --almost-all
    do not list implied . and ..
  --author
    with -l, print the author of each file
  -b, --escape
    print C-style escapes for nongraphical characters
  --block-size=SIZE
    scale sizes by SIZE before printing them; e.g., --block-size=K prints
    sizes in units of 1,024 bytes; see SIZE format below
  -B, --ignore-backups
    do not list implied entries ending with ~
  -C
    with -l: sort by, and show, ctime (time of last modification of file sta
  Press q to quit, h for help, and SPACE to scroll.
```

► This is the man page for **ls**, and it helpfully lists the various options that you can use in alphabetical order.

Desktop viewing

Man and **info** are intended to be readable in a terminal, because you may need to use them on a system without a desktop. There are GUI viewers for them, though, the most convenient being in KDE, where you can press [Alt]+[F2] and type **man:/command** or **info:/command** and get an HTML formatted version of the document displayed in *Konqueror*. There are also the *tkInfo* and

tkMan programs to display the respective formats in a GUI window.

There are several websites containing comprehensive man page collections: such as <http://linux.die.net>, www.linuxmanpages.com and <http://manpages.ubuntu.com>. These are particularly useful if you want to read about something that you do not have installed locally.



► KDE users can read **info** and **man** pages in a browser, with clickable links, thanks to KDE's KIO slaves.

As a normal user, you would normally only use sections 1, 5 and 8 (and possibly 6). If you use the section number with the **man** command, it will only look in that section, otherwise it will show the first match it finds. This is necessary because you can have more than one page with the same name. The **passwd** command is used to set user passwords, which are stored in the file **/etc/passwd**. Try:

```
man passwd
man 1 passwd
man 5 passwd
```

The first two document the **passwd** command from section 1, while the third shows the man page for the **passwd** file. This is one of the strengths of the man page system – it documents everything: commands, configuration files, library functions and more. It's not limited to specific commands or files, and section 7 contains man pages for all sorts of things. Ever wondered how symbolic links work, or what happens when you turn on your computer? Try:

```
man 7 symlink
man 7 boot
```

Quick help

There is more to **man** than a bunch of formatted text pages and a program to display them in a pager. **Man** maintains a searchable database of pages, which is automatically updated by **Cron**, and has some other programs for working with it. Each man page has a NAME section, which includes a short description of the page's topic. The **whatis** command gives the description – it tells you what the program (or file) is, without going into details of options. Here is the classic geeky, recursive example:

```
whatis whatis
whatis          (1) - search the whatis database for
complete words
```

This is a faster way to see what commands do, especially if you want to check more than one:

```
whatis grep sed symlink
```

The **whatis** command searches only the name, and only matches on whole words, so it assumes you know the name of the command and just want to know what it does. For a more wide-ranging search, use **apropos**, which does a similar job but searches the descriptions as well and returns all matches – compare these two commands:

```
whatis png
apropos png
```

There is another form of documentation that's favoured by the GNU project: info pages. While a man page is basically one very long text file with a bit of formatting markup, an info document contains a tree of linked pages in a single file. It's more like HTML than plain text, but is designed for reading in a text console and all the 'pages' are contained in a single file.



Unsurprisingly, the command used to read info pages is this:
info info

This time, the self-referencing is not gratuitous. Man pages are usually quite intuitive to navigate – it's just like reading any other text in a pager. **Info** uses a different set of key commands, so reading its own info page is a good place to start. You move around individual pages as you would normally, but if you press [Enter] when the cursor is on a link (noted by an asterisk), you descend into that node, getting further information on the topic at hand. To go back up to the parent node, press [u]. You can also navigate within a level of the documentation tree with [n] and [p], which take you to the next and previous nodes. If you have ever looked at any GNU documentation online (<http://bit.ly/grubmanual>) you will recognise this layout. It's simpler than HTML, with the navigation commands generally moving around the same level of the tree or up and down one level.

You may be asking yourself what the point of this structure is. Well, if you've ever tried to find information in a long man page, such as **man bash** or **man mplayer**, you'll know how frustrating and time-consuming the 'everything on one page' approach can be. Info documents are divided into sections and chapters, enabling clearer and more succinct presentation. The majority of GNU programs have fairly brief man pages, but have more detail in their info pages. Splitting the document up into pages alters the way that searching is carried out. Press [s] followed by a search string and then [Enter]. Info will jump to the next occurrence of the string, even if it's in a different node. Continue to press [s] then [Enter], with no search string, to jump to subsequent occurrences of the same string. Those keys, plus [q] to quit, should enable you to navigate info pages with ease. You might be wondering why we're not using HTML. The main reasons are that **info** pre-dates HTML, and that info documents are contained within a single file. Conceptually, they are similar – so much so that **info2html** (<http://info2html.sourceforge.net>) can be used to convert an info document to a series of HTML pages.

► The **ls** info page goes into more detail and groups options according to their function. Info pages generally provide more detail than man pages.

Printing manuals

There may be times you want a hard copy. As **man** pages are stored in a markup format and converted for display by the **man** program, you can use **-t** to convert them to a printable format, Postscript by default, like this:

```
man -t somecommand | lpr
```

The Postscript is output to **stdout**; piping it to **lpr** like this sends it straight to the printer. You could also convert the Postscript to PDF, and therefore create versions of your man pages suitable for

carrying around on a tablet or e-reader:

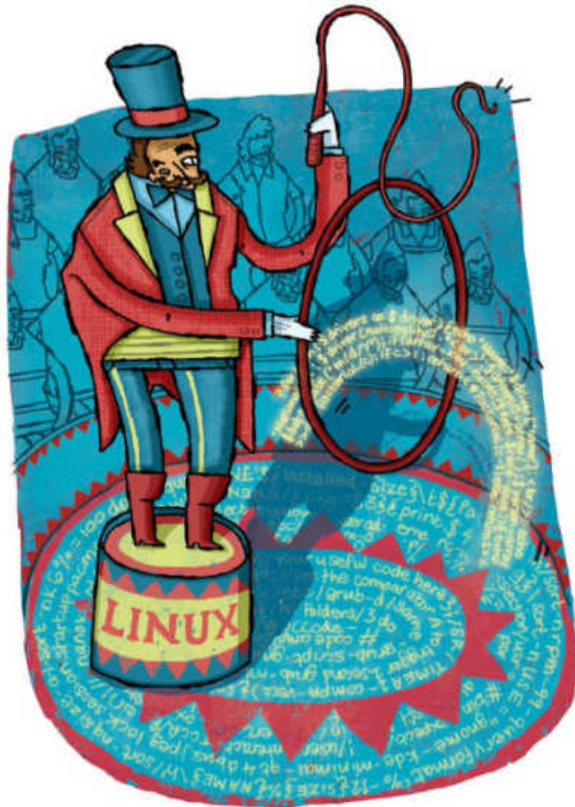
```
man -t somecommand | ps2pdf -
somecommand.pdf
```

Print info documents by passing them through the **col** command:

```
info somecommand | col -b | lpr
```

Sysadmin: Back to basics

You need some basic admin skills under your belt if you want to be able to use Linux effectively. Read on and refresh your memory...



Since you've read this far, there's a high probability that you've already set up your Linux distribution. You've probably also used the distro's package manager to pull-in additional software, and are capable of browsing the web and perhaps playing games. While you might get some fair mileage with your current setup, sooner or later you'll need to tweak things, whether you want to install a new piece of hardware, troubleshoot the performance of an app or share your computer with others.

Most distributions bundle tools to help users manage various aspects of their system. Some distros, such as Ubuntu, have fairly easy to grasp administration tools, while others have quite an expansive administration section with tools that aren't very intuitive at first glance. The average home user doesn't need to have the proficiency and the dexterity of a professionally qualified system admin, yet there are a few skills that should be in your repertoire to help you manage your computer efficiently.

Quick tip

By using various combinations of permission settings, you can quickly set up a more secure environment.

Managing users is one of the most basic Linux admin skills you should possess, even if you are the only person using the computer. That's because knowledge about users and groups is imperative in order to secure and control access to data. Although you can manage users from the command line, almost all modern Linux distros include a graphical tool to help you add, control and remove user accounts. However, there are a few things you should keep in mind when managing users and groups.

It may sound obvious, but a user is anyone who uses a computer. Some system services also run using restricted or privileged accounts. The superuser, root, has complete access to the operating system and its configuration. Unprivileged users can use the **su** and **sudo** commands for controlled privilege escalation.

Behind the scenes, a username has a unique string of numbers known as a user ID (UID). Similarly, groups have group IDs (GID). The root user is UID 0. Numbers from 1 through to 499 – and number 65,534 – are the system users, sometimes called logical or pseudo-users. Regular users have UIDs beginning with 1,000. When creating a user, the system admin can add other users to an existing group, in which case they'll all bear the same GID. Alternatively, they can create a new group, in which case the GID will probably be the same as the unique UID for each user. Remember that a group cannot be a member of another group in Linux.

Files and permissions

Under Linux, everything is a file, including directories and devices. Every file has an accompanying set of ownership permissions, which provide data security by giving specific permission settings to every single item. They control who can read, write or execute a file. These three access permissions are set individually for three types of users: the file's owner, the members of the group the file belongs to, and all others on the system.

To list the permissions, type in this command:

```
ls -l
```

When used without specifying a filename, it will list the permissions of all files within the directory in this format:

```
-rw-r--r-- 1 bodhi bodhi 8980 Apr 24 09:21 somefile.txt
drwxr-xr-x 9 bodhi bodhi 4096 May 29 07:42 Downloads
```

The first character in the output indicates the type of file. The hyphen in the truncated listing above represents a plain file and the **d** represents a directory. There are other characters as well to identify the different types of devices. This is followed by the file's read, write and execute

permissions for the owner, group and all others on the system. The number following this indicates whether it is just a single file (1), or the number of additional hard-linked files. A hard-linked file is an exact copy of the file that might be located elsewhere on the system.

Next up is the name of the owner (the user who created the file), and the group whose users can access the file. The next pieces of information indicate the size of the file in bits, and the date when it was last modified.

Let's get back to file permissions. As you can see in the listing, permissions are indicated by three characters: **r** for reading, **w** for writing and **x** for executing a file or reading a directory. The permissions are grouped for owner, group and others. The owner of `somefile.txt`, `bodhi`, can read and write (**rw**) the file, while other members of the group `bodhi` – as well as all other users – can only read the file (**r--**).

Many users prefer to use numeric codes, based on octal (base 8) values, to represent permissions. Read permissions are assigned a value of 4, write gets 2, and execute gets 1. Applying this to the listing, the file has a permission of 644, because the user with read (4) and write (2) permissions gets a value of 6, while the group members and other users with read permissions get 4 each.

You can alter the permission of a file, using either form of permission notation, with the **chmod** command. There are several options if you wish to use the mnemonic forms with **chmod**. The three permissions for the owner are controlled with **u**, while **g** controls the permissions for the group, and **o** controls them for others not in the group. The character **a** influences all permissions for all users, while the **r**, **w**, and **x** characters control the read, write and execute permissions. You can use any of these with either the plus sign to add a permission, or the minus sign to revoke permission for a file.

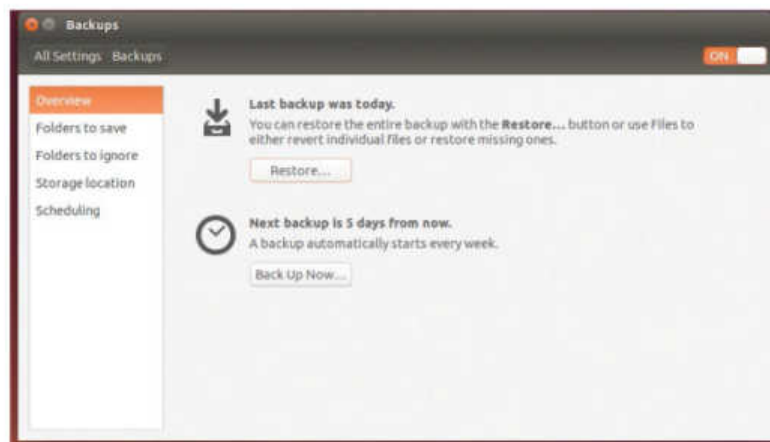
The command listed here, for example, will give read and write permissions to the owner of the file:

```
chmod u+rw somefile.txt
```

However, you are advised to use the octal values to alter the permission, which is easier to comprehend. The above task can also be accomplished with this command:

```
chmod 600 somefile.txt
```

Here, we have given the owner permission to read and



» Tools such as *Déjà Dup* help take the sting out of complex and involved tasks such as backups.

write to the file while denying permissions from other users. Then there's the **chown** command, with which you can transfer ownership of a file. The command **chown chris somefile.txt** hands the file to user `chris`. Similarly, you can also move a file from one group to another with the **chgrp** command. For example, this command will move the `index.htm` file to the *Apache* web server's `www-data` group:

```
chgrp www-data index.htm//
```

Compile software

Compiling software is another skill you need in your arsenal. Although most popular software is available as pre-compiled binaries via the distro's package manager, sooner or later you'll run into some that you'll have to compile yourself. It could be a bleeding edge version that is critical to your workflow but hasn't been packaged by your distro yet.

Compiling software from source isn't difficult and only slightly more involved than installing binary packages. But make sure you have all the tools required to compile packages. If you are using a Debian-based distro, such as Ubuntu, you can install the build-essential package with this:

```
sudo apt-get install build-essential
```

Users of RPM-based systems, such as Fedora, will need to grab the individual components with this command:

Quick tip

If the compile happens to fail, check the error messages and then run the **make clean** command before you start again. Run **make uninstall** as the root user to remove any software that you have installed from source.

Restore the bootloader

If you dual-boot Linux, there's a risk that the other operating system may replace your distro's bootloader with its own. The good news is that you can restore the *Grub2* bootloader – the default for a majority of distributions – with little effort.

The recommended way is to boot off your distro's Live CD and open a terminal window from inside the Live session. Now figure out the device on which you have installed the distro. Use this command to figure out the device with the Linux installation:

```
sudo fdisk -l command
```

Once you have this info, issue the following command in the terminal:

```
sudo grub-install /dev/sdX
```

Make sure that you replace the **X** in the command with the letter that represents your disk, such as **sda** or **sdb**. This should reinstall the *Grub 2* bootloader in the master boot record of the disk. When it's done, the normal *Grub* boot menu should appear when you restart the computer.

Alternatively, you can use the graphical *Boot-Repair* tool by installing it inside the Live environment using the recommended method for that particular distro. Once the tool is installed, fire up the tool, which then scans your system before it launches. When it's running, hit the big 'Recommend Repair' button, which should fix the boot loader.



» The *Boot-Repair* tool offers the easiest option to fix issues within the *Grub* bootloader.

```
yum install make automake gcc gcc-c++ kernel-devel
```

The source code of most software is available as compressed tarballs. These are TAR files that have been compressed with *Gzip* and sport the *.tar.gz* extension. Download the source code into your home directory. You can uncompress it with the **tar** command:

```
tar zxvf softwaresource.tar.gz
```

Now move into the extracted directory and look for a file that's named **Install** or **Readme** or something along those lines. This text file contains specific instructions about how to compile that particular piece of software. The generic procedure to compile software begins by calling a configuration script with the **/configure** command, which checks whether or not your computer meets the specific requirements of the software. If you are missing any dependencies, the configure script will abort, telling you the name of the missing components. You can then install these via the package manager.

Once the configure script has run through its checks successfully, you can compile the software by issuing the **make** command. This command will take some time, depending on the size of the software you are installing. When it's finished, round up the procedure by running the **make install** command as the root user.

Navigate the file system

You should already be familiar with your user directory, which is under the **/home** directory. To be an effective Linux user, you should also familiarise yourself with the filesystem and its organisation.

The commands essential to keep the system up and running are kept under **/bin**. Unlike regular apps that are dynamically linked and depend on shared software libraries to run, the commands under **/bin** are statically linked and do not depend on software libraries residing elsewhere.

The **/etc** directory houses the system configuration files and directories. While some software packages such as *Apache* and *OpenSSH* have their own directories within **/etc** that contain their configuration files, others such as **fstab** only use one configuration file. An important directory under **/etc** is **modprobe.d/** which contains instructions for loading kernel modules during system startup.

The **/proc** directory differs from the others in that its contents are created from memory and exist only while Linux is actually running. You can get information about the computer by reading the contents of files under this directory. This command, for instance, will display various details about

the CPU:

```
cat /proc/cpuinfo
```

Then there's the **/usr** directory, which contains apps, libraries and other types of shared data for use by anyone on the system. The **/var** directory contains directories used by various system services for spooling and logging. Some of these, such as print spooler queues under **/var/spool**, only exist temporarily. Others, such as the logs under **/var/log**, are renamed and rotated regularly.

Talking of log files, checking on them is an essential admin task. The Linux kernel, critical network services and other important apps all record critical activity inside log files that are filed under the location **/var/log**. In particular, two important log files you should regularly look at are **auth.log**, which logs all authentication attempts made on the system, and **boot.log**, which records messages generated while the system boots and starts services.

Then there's that **/home** partition that you are already familiar with. It is very good practice to keep **/home** on a separate partition from the rest of the filesystem. This ensures that the data inside will survive any damage to other parts of the distribution.

Monitor the system

Another important aspect of a system admin's job is to make sure the computer they are looking after continues to run smoothly. If you want to check how things are doing, you can use the **ps -e** command. This will print a long list of all the processes that are running on the system. You can truncate the list to display only processes that have the same UID as the current user by invoking the **ps** command without any switches. The output of the **ps** command includes the unique process ID (PID) along with the name of the running program.

It's more common to pipe the list to display information about a specific program, however. This command will show information only about the processes that are associated with *LibreOffice*:

```
ps aux | grep libreoffice
```

Once you know the PID of a process, you can terminate that process with the **kill** command. Assuming that *LibreOffice* has a PID of 8899, you can terminate an unresponsive session with this command:

```
kill 8899
```

You can use the **top** command to view a list of running tasks with the most CPU-intensive ones listed first. The **top** command displays various bits of information about the processes, including its PID. It also has a few interactive commands. For example, you can kill a process from within **top** by pressing the [K] key, which prompts you to enter the PID of the process you want to terminate.

If you have a process that's hogging the resources on your computer, the **renice** command will assign it a lower priority. Linux assigns a priority to each process, and the ones with higher priority are given access to the system's resources first, while others that have lower priority need to wait for their turn.

The **renice** command can assign a priority value between -20 and 19. The lower the number, the higher the priority. For example, if you wanted to take away resources from *LibreOffice* (PID: 8899) you could assign it a lower priority number with this command:

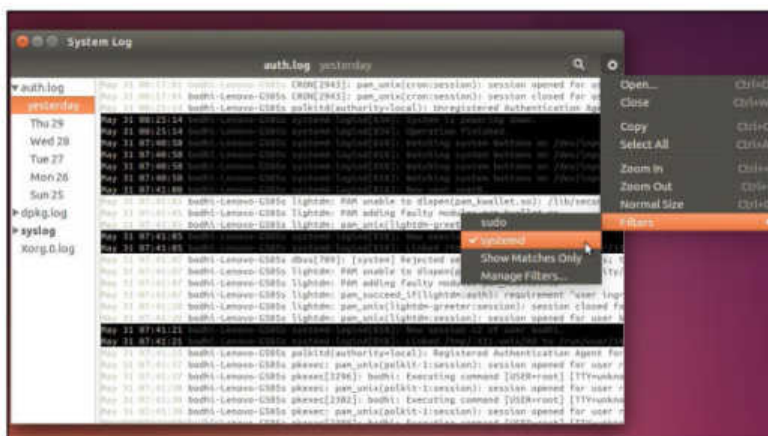
```
renice 15 8899
```

If the command line isn't really your cup of tea, the Gnome and KDE desktop environments also include their own system monitoring tools in your distro's official repository.



Quick tip

Ensure that you always make a backup of a configuration file before you start to edit it by using the **cp** command, such as **cp configfile configfile.original**.



Some distros include a graphical tool for displaying logs.

Make regular backups

Although a Linux distro is pretty stable and more immune to common malicious programs floating around online, the hardware it runs on and the operator are not infallible. Therefore, it's good to implement a back-up strategy.

The average desktop user should back up their home directories as well as any configuration files. A weekly backup should suffice for most desktop users. You can store several weeks of backups on an external USB drive. Once you start running out of disk space, you can either discard the older ones or burn them on an optical disc.

There are several open source back-up tools with varying degrees of complexity and features. The command line **tar** is arguably the most popular Linux back-up utility. You can use it to make compressed archives of entire directories. This command will create an archive of the Downloads directory:

```
tar cvf downloads.tar ~/Downloads
```

To compress the backup, modify the command to:

```
tar cvfz downloads.tar.gz ~/Downloads
```

The only difference between the two commands is the addition of the **z** option, which asks **tar** to create a **gzip** compressed archive. The resulting filename changes, too.

Depending on the size of the directories that you are backing up, **tar** may take some time to create the compressed back-up file. If you have large amounts of data that needs backing up, it is often more useful to make incremental backups, which will only back up those files that have changed since the last time one was carried out.

Before you can create an incremental backup, you need to first take a full backup of the intended directory using the command listed above. Next, you'll have to create a special file that contains a list of all the files included in the backup:

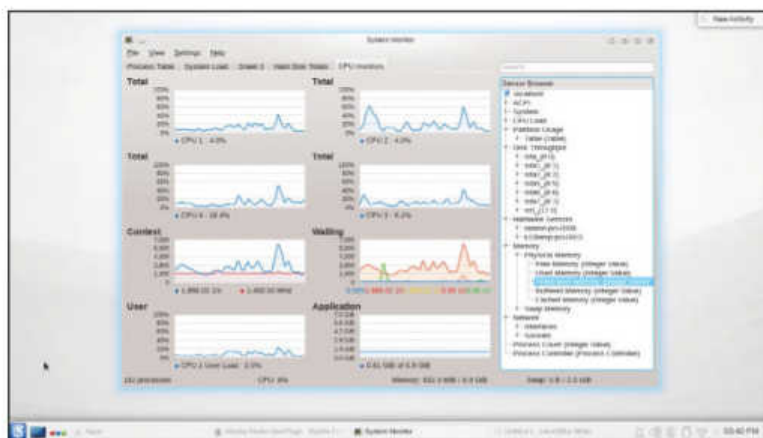
```
tar cvzg increment-file -f downloads.tar.gz ~/Downloads
```

When you return to back up the folder in a week, use **increment-file** to only back up files that have changed.

The following command creates a new back-up file that is considerably smaller than a full backup because it contains only the changed files:

```
tar cvzg increment-file -f 31052014-downloads.tar.gz ~/Downloads
```

Use this last command to create incremental backups every week. Remember to use a different name each time you



» KDE's system monitor can visualise all kinds of resources inside the computer.

create an incremental back-up file.

When you want to restore data from these back-up files, begin by extracting the full back-up file, followed by the increment files in chronological order. This means that after extracting the full back-up file, you'll extract the first incremental back-up file that you created, and end with the latest incremental back-up file, like so:

```
$ tar zxvf downloads.tar.gz
```

```
$ tar zxvf 31052014-downloads.tar.gz
```

```
$ tar zxvf 07062014-downloads.tar.gz
```

The contents of the folder will then be restored as they were when you created that last snapshot. If you aren't interested in creating incremental backups, and are looking for an easy-to-use tool, check out the graphical *Déjà Dup* app. The app has a minimal interface and doesn't overwhelm new users with very many options. It is used as the default back-up tool on several distros, including Ubuntu.

To use *Déjà Dup*, mark the folders you wish to back up and specify where you want them to be saved. This can be on the hard disk, or a remote location that you can connect to via SSH, FTP or *Samba*. You can also set a schedule and the app will back up the selected folders in the background.

While this tutorial is by no means an exhaustive list of a system administrator's role, you should now have the skills required to take adequate care of your computer. If you need more detail, jump back to page 104 for our Terminal series.

Schedule tasks

There are some admin tasks – creating backups and large downloads, for example – that are far more effective when you execute them as per a schedule.

The **at** command helps you schedule tasks that you need to run at a specific time and date. For example, if you need to download a large file, it's best to schedule it late in the night or very early in the morning. The command **at 1am tomorrow** will change the prompt to **at>** and everything you type at this prompt will be run at the specified time. To download a file, point to its location with the **wget** command. Press the [Enter] key to specify another command – for example, if you want to move the file to a specific folder. When you are done, press [Ctrl]+[D] to save the job. At the designated time, the **at** command will perform the actions specified. The **at** command allows fairly complex time

specifications. In addition to AM and PM, it also accepts times in the HH:MM format and recognises particular dates as well.

When you press [Ctrl]+[D] to submit a job, the command prints a job ID. Use the **atq** command to list all submitted jobs, which you can then remove with the **atrm** command by suffixing the ID of the job you wish to delete.

If you wish to run a task repeatedly, you're better off using the **cron** daemon. Use this command to edit your crontab file:

```
crontab -e
```

The first time you run the command, you will be asked to select one of the available command line text editors.

Each job is specified in seven fields that define the time to run, the owner, and the command. The first five commands specify the execution time, the sixth defines the day of the week, and

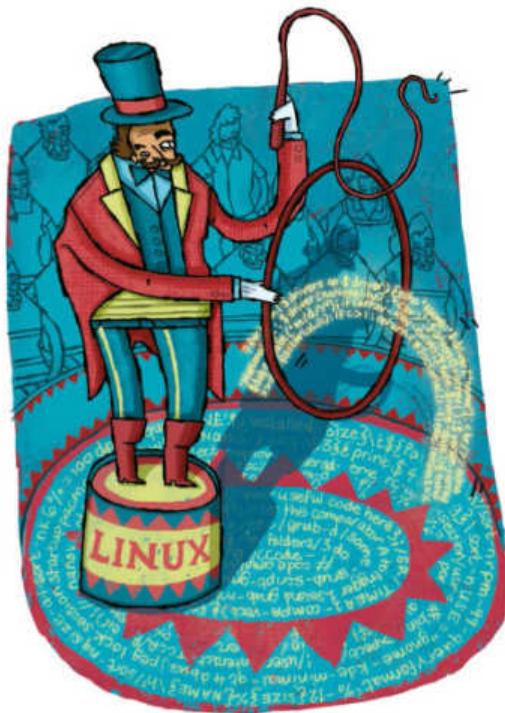
the last field lists the command to be executed. You can use the **crontab -l** command to list your scheduled jobs. System-wide crontabs are stored in **/etc/crontab**, whereas user-specific crontabs can be found under the **/var/spool/cron** directory.



» You can find utilities on the internet which help you to set up crontabs easily.

Sudo: Control root access

If you've come this far it's time to revel in the god-like powers of the superuser, using sudo to grant the wishes of lesser beings.



Linux keeps the administrative user separate from any of the other users on the system. Even if you are the only user on your computer, there is still a root or superuser account that has to be used for certain tasks. Some distros, such as Ubuntu, attempt to hide the superuser account, you cannot log in as root but it's still there – the system would not work without it. So why do you sometimes

need to work as root, how do you do it and how can you allow other users to perform root tasks without giving them carte blanche with your system?

The traditional way to give root access is with the **su** command. This command, which stands for switch user, not superuser, enables you to become another user for as long as it is running. When run in a shell with no arguments, it lets you into the root account (provided you know the root password). It is not exactly the same as logging in as root, you still retain some elements of your current environment. To provide as close as is possible to a login environment, add the **-l** option. We said the **su** stands for switch user as it defaults to switching to root, but you can give a username to the command, eg:

```
$ su -l fred
```

Once you have run **su**, you will stay as that user until you type exit, or press Ctrl+D. This is not necessarily a good thing, leaving a shell open as root is not good security practice, so you can pass it a single command to run with the **-c** option:

```
$ su -c "ifconfig eth0 up"
```

This doesn't open a shell session, instead, after asking for the root password, it runs the command given.

Then came sudo

The traditional **su** has some drawbacks: it requires the user to know the root password, it then lets them run anything they like and they can leave a root shell running after they have done what they need. The **su -c** method is clumsy if you want to run more than one command, requiring the root password for each one. The answer to all of these is **sudo**, which most distros use nowadays.

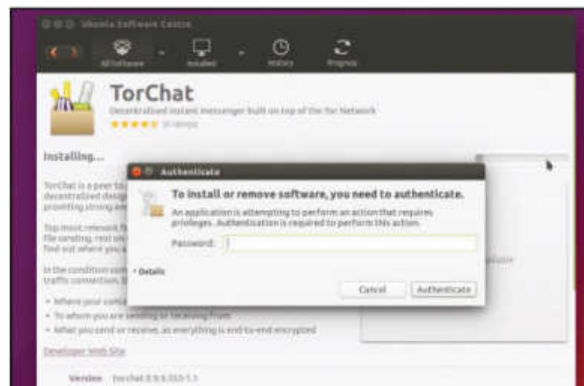
The **sudo** command works differently. First, it's used to run a single command, such as:

```
$ sudo ifconfig eth0 up
```

Note: We don't need to enclose the command in quotes now. Second, it asks for your passwords, not the superuser's, so there is no need to give anyone the root password (in fact, Ubuntu doesn't have one by default). Try running another command with **sudo** and you will see another difference, it doesn't ask for your password again. It will also allow you to run another command within five minutes of the previous one without entering a password. That timeout can be changed in the configuration file if you wish.

If the user only needs to give their own password to run a

» The various desktops have UI versions of **sudo**, which are used when running root commands from the desktop.



command as root, what is to stop any user running any command as root? The answer lies in the configuration file at **/etc/sudoers**. This specifies exactly who can do what, giving fine-grained control over execution permissions. The default configuration is to allow members of the wheel group to execute any command. Some distros change this to a different group name, such as admin (wheel – as in you're a big wheel – is a traditional name for a group with elevated privileges that predates even Unix). Most installers make the user created during installation a member of that group, subsequent users are not unless added specifically. This means the user that installed the distro is automatically given admin privileges, without the need for anyone to open a root terminal or even know the root password.

visudo not vi

You can grant **sudo** privileges to other users by editing **/etc/sudoers**, but you should never do this manually. If you were to edit this file and introduce a syntax error, you could potentially stop **sudo** working. As you need it to edit this file, you need to be root to even read the file for security reasons, you would not be able to undo your mistake. The answer is the **visudo** command, which creates a temporary copy of **/etc/sudoers** and loads it into your editor. After you have edited the file, **visudo** checks that it's correct before overwriting the real **/etc/sudoers** file. The name may imply that this uses **vi** to do the editing, but fear not, that's only the fallback option. In fact, **visudo** uses your default editor, which may be defined in the **EDITOR** environment variable, or the system-wide setting in the alternatives system on Ubuntu and other Debian-based distros.

To change the system default editor, run:

```
$ sudo update-alternatives --config editor
```

You can set the **EDITOR** environment variable in your profile or choose it on the command line, like this

```
$ EDITOR="gedit" sudo -E visudo
```

The **-E** is needed to make sure the environment variable is passed through to the root user, more on environment variables later. Now we know how to edit the **sudoers** file, let's see what we can do with it. The file contains two main types of entries: aliases, which are basically variable definitions, and user specifications. The latter entry sets who can do what, eg the default file contains a line like:

```
%wheel ALL=(ALL) ALL
```

The basic format of this line is "who where = (as_whom) what". The **%** at the start of the name means this is a group and applies to all members of it. The 'where' is the hostname of the computer and the next item states which users the commands may be run as. The last item is a list of commands that can be run. **ALL** is the **sudo** wildcard, so this line means that members of the wheel group can run anything anywhere, a standard entry for the admin user. If you want to give the same privileges to another user, the simplest option is to add them to the wheel (or admin, depending on your distro) group:

```
$ sudo gpsswd -a george wheel
```

Restricting root access

What if you don't want george to be able to do anything, but you want him to be able to edit web files, owned by the **Apache** user? You could use something like

```
george ALL=(apache) ALL
```

Logging in as root

Although the root account is disabled in some distros, you can work around this by running one of the following:

```
$ sudo bash
```

```
$ sudo --login
```

If you want to prevent this, you can use **!** to negate commands that cannot be run by the user to, eg:

```
%wheel ALL=(ALL) ALL, !/bin/bash,!/bin/sh,!/bin/zsh
```

Why would you want to do this? One reason is that every command run with **sudo** is logged, making it easy to see who did what on a system with multiple administrative users. Switching to a root shell, as above, would negate that.

There's no leading **%**, so this is a username. We are still allowing all commands on all hosts, but only as the **Apache** user

```
$ sudo emacs index.html
```

would give an error, because **sudo**, like **su**, defaults to the superuser if none is specified. This would work because it's run as the allowed user.

```
$ sudo -u apache emacs index.html
```

Of course, it still allows george to run any non-root command, although the scope in which he can run them is severely restricted. If we want to restrict his activities to editing files, we can do so like this:

```
george ALL=(apache) /usr/bin/emacs, /usr/bin/vi
```

We have replaced the final **ALL** with a list of commands, using the full paths and separated them by commas. We can also specify allowable arguments, so:

```
george ALL=(ALL) /bin/mount /dev/sd[c-f]*, /bin/umount /dev/sd[c-f]*
```

This will allow george to mount and unmount filesystems, but not on **sda** or **sdb**, so he can work with removable drives. There's one more important addition to this line (although there are many more covered in the documentation), the **NOPASSWD** option means the user will not be prompted for their password – use with care!

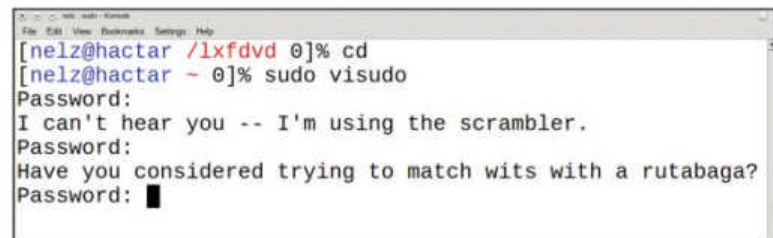
```
george ALL=(apache) NOPASSWD: /usr/bin/emacs, /usr/bin/vi
```

Environment variables

Linux uses a number of environment variables to keep track, such as **\$HOME** and **\$DISPLAY**. The default behaviour of **sudo** is to clear most of these for the execution of the command. Which may not always be what you want. A brute force way around this is to run **sudo** with the **-E** option to keep them all. A more structured solution is to specify the ones you want to keep in **sudoers**, eg you may need this to be able to run GUI programs with **sudo**:

```
Defaults env_keep += "DISPLAY HOME"
```

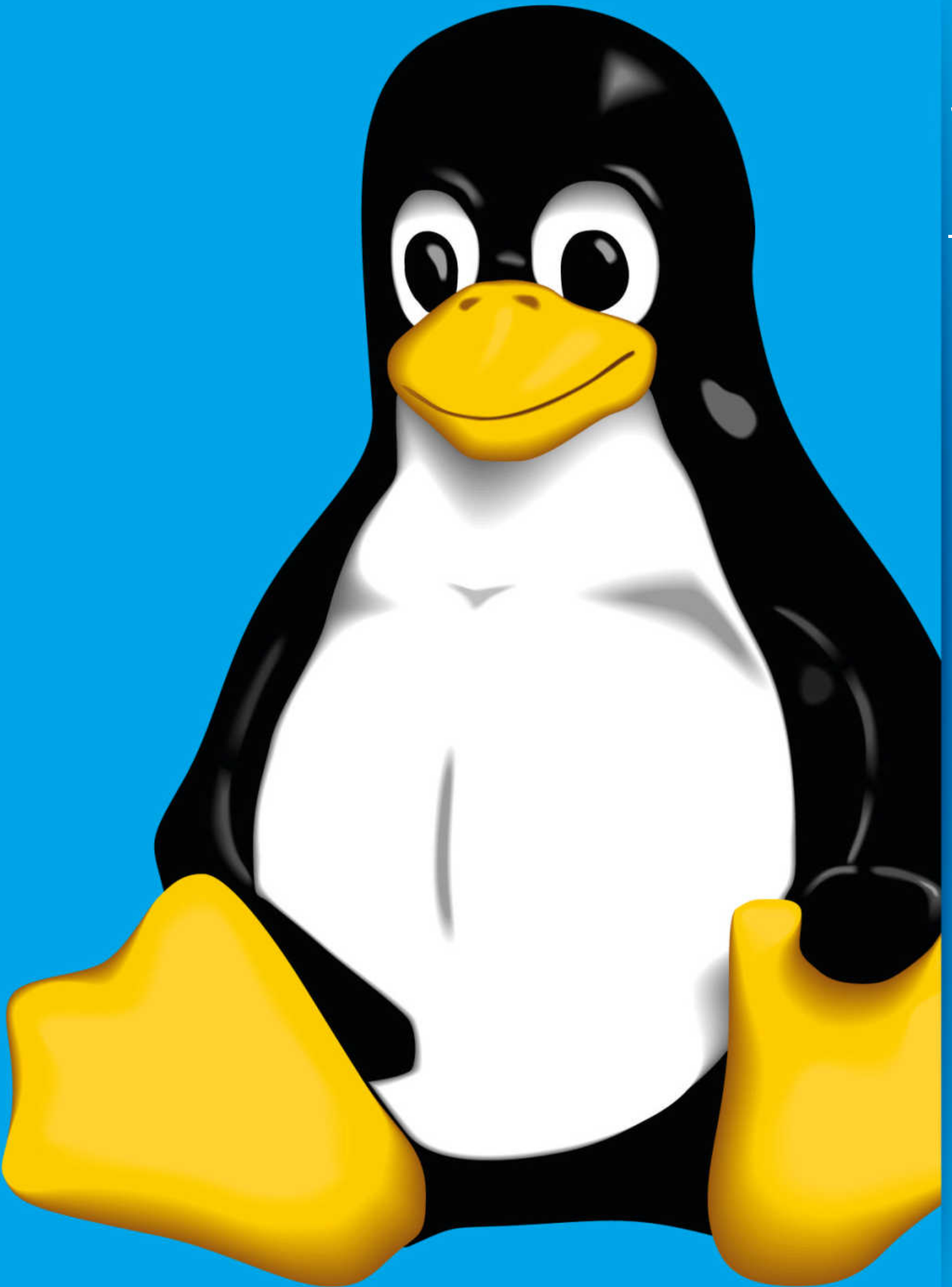
There are various other settings you can add to **/etc/sudoers**, all of which are thoroughly documented, but this should be more than enough to get you started.



➤ The default configuration includes throwing mild insults at you if you give an incorrect password. You can turn this off for a more professional appearance.

Projects

Build your own router	128
Try a microkernel OS	132
Repair broken systems	136
Analyse network traffic	140



Router: set up a gateway

Think you can do better than your own router? Let's throw down some iptables rules and make our own hotspot.



```
File Actions Edit View Help
Shell No. 1
[root@jbmachine jonni]# systemctl start hostapd
[root@jbmachine jonni]# brctl show br0
bridge name      bridge id      STP enabled     interfaces
br0              8000.0013efc70016  no              wlp0s11flu4

[root@jbmachine jonni]# brctl
Usage: brctl [commands]

commands:
addbr          <bridge>          add bridge
delbr          <bridge>          delete bridge
addif          <bridge> <device>    add interface to bridge
delif          <bridge> <device>    delete interface from bridge
hairpin       <bridge> <port> {on|off}  turn hairpin on/off
setageing      <bridge> <time>        set ageing time
setbridgeprio  <bridge> <prio>        set bridge priority
setfd          <bridge> <time>        set bridge forward delay
sethello       <bridge> <time>        set hello time
setmaxage      <bridge> <time>        set max message age
setpathcost    <bridge> <port> <cost>    set path cost
setportprio    <bridge> <port> <prio>  set port priority
show           [ <bridge> ]          show a list of bridges
showmacs       <bridge>          show a list of mac addresses
showstp        <bridge>          show bridge stp info
stp            <bridge> {on|off}    turn stp on/off

[root@jbmachine jonni]#
```

► A bridge too far. We're not meant to mess with Future Towers' networks, but brctl makes it easy.

just be paranoid and want to install some additional firewalling for a small internal network: you could only allow certain traffic, or force all traffic to be routed via a VPN. You can even install *Wireshark* on the gateway machine and check out how much Internet chatter comes out of the devices connected to it.

We'll cover setting up a wireless gateway later, but for now assume we've got some computers that aren't connected to the external network, and one computer (the gateway) that is. We'll assume all of these computers are connected by wired connections to a switch and that the gateway machine is happily talking to the internet (either wired or wirelessly – for now we're just assuming that the internal network isn't connecting wirelessly to the gateway, this requires some extra configuring that we'll cover later).

Setting up IP addresses

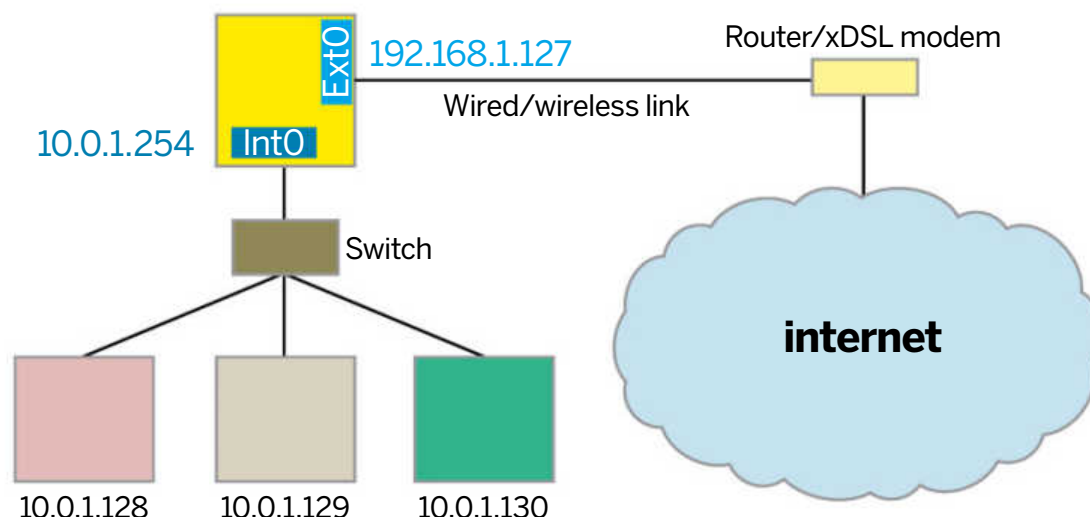
The first thing we need to do is get the machines talking to each other, which requires getting their IP addresses set up nicely. We won't deal with IPv6 in this tutorial (but soon my pretties), since most home routers still work with IPv4 and it's simpler to keep our protocols homogenous. Your distro may have progressed to kernel-generated persistent names for your network devices, in which case your wired and wireless devices will, have names such as `enp0s327` and `wlp999`, respectively. Or you may still have the old-style, human-readable names, such as `eth0` and `wlan0`. We'll just refer to

You probably have at least one router in your house. They perform the not insignificant task of routing data from one network, eg the internet, to another, such as your home wired/wireless network. While in theory you could build on this tutorial and replace the box your ISP gave you, it's probably not the best idea – getting a Linux box with a DSL or ADSL modem in it to talk to your ISP can be tricky. Besides, many of these hubs already run Linux, and many more allow you to install a DD-WRT or OpenWRT image, both of which handle much of the ugliness for you. But there's lots of other routing that's useful...

Suppose your wireless card breaks, or needs some new firmware, leaving you without connection or a long enough patch cable to reach the router. Certainly you could replace the broken hardware or download the files from another machine, but that's hardly cricket. Instead, why not have a handy, portable, lifesaving gateway machine around, that can, via a simple crossover cable, feed the starving machine the network it so desires? You could even connect several machines this way via a simple switch – only the gateway machine needs to be able to see the internet, or whatever external network you want to share access to. You may also

Quick tip

Simple traffic routing doesn't require a powerful CPU, but if you're anticipating a lot of traffic you'll want to make sure you've a Gigabit adaptor in the gateway and a gigabit switch.



Thanks to the miracle of NAT, packets can traverse the murky marshes of our internal network, negotiate all kinds of perils in the 192 plains and soar amongst the cloud birds of the wider internet.

the external interface of the gateway (eg the one connecting to your ISP-supplied router) as `ext0`, and the internal interface (the one connecting to the switch) as `int0`. So you'll have to make your own substitutions.

RFC1918 provides a few standard IPv4 addresses that can only be used for internal networks, eg `10.x.x.x`, `172.16.x.x`, `192.168.x.x`, so let's go with the first format. We can be a bit more specific here, so let's stipulate that all our internal IPs are in the form of `10.0.1.x`. This is usually written `10.0.1.0/24` ie a 24-bit netmask (`255.255.255.0`). We'll set up the gateway machine's Ethernet interface manually, with the static IP address **10.0.1.254**. Depending on how your machine is set up, this might be achieved through *Network Manager*, the **ifconfig** or **ip** commands, making a *netctl* (Arch Linux) script, editing `/etc/network/interfaces` (Debian). Whatever you choose, it should be straightforward, and also not interfere with the gateway machine's existing internet connection.

We could do this for all the machines on our network, but it's easier to use the *dnsmasq* program, which provides a simple DHCP server to allocate addresses. This program will also be useful later since it will enable the gateway machine to masquerade as a DNS server (hence the program's name).

Installing *dnsmasq* will just be a matter of `$ sudo apt-get install dnsmasq` or whatever is your distribution's equivalent command is. This will install a heavily-commented `/etc/dnsmasq.conf` file which we shall tweak to our requirements. It's good practice to add the following lines right after their commented equivalents, so that options are sensibly grouped. We'll first set up our ersatz DNS server to listen to requests from our fledgling internal network. Add the following line to `/etc/dnsmasq.conf`:

```
listen-address=10.0.1.254
```

We'll also set up our DHCP server while we've got this file open. It's going to allocate addresses in the range **10.0.1.128** to **10.0.1.253**, this way addresses with a lower final octet can be reserved for machines on the internal network which need static IPs. It's also possible to have DHCP assign specific addresses to specific machines based on their MAC address, using the **dhcp-host** option:

```
dhcp-range=10.0.1.128, 10.0.1.253, 12h
```

Now start the *dnsmasq* service with

```
$ sudo systemctl start dnsmasq
```

or if you're not running *systemd*:

Quick tip

Basic forwarding, routing and masquerading can now be carried out via *systemd*'s networking daemon, *networkd*. Is nothing sacred? We're taking bets on what its next trick will be. Our money's on *Sirystemd*: "Sir, can I have my init scripts back, please?" No! – ED.

Hotspots with hostapd

On Debian (and friends) `sudo apt-get install hostapd` will install, besides the *hostapd* program, a gzipped sample configuration file which you can peruse with:

```
$ zcat /usr/share/doc/hostapd/examples/hostapd.conf.gz | less
```

Other distros will install a similar file, probably in another place. This file is heavily commented and gives examples for setting up all manner of access points. We'll only need a few options to get our hotspot up and running, so rather than edit the example file, we'll start afresh. *Hostapd* needs to know where to find this file so **edit** `/etc/default/hostapd` and add the line:

```
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

This is a sane place to put the file and may already be the default on your distribution.

Before we proceed any further, make sure that you know the name of your wireless interface (you can check this with **ifconfig**). For this section we'll refer to the wireless network as **wlan0** and the wired external network as **eth0**. A basic WPA2 password protected network is set up with the following `/etc/hostapd/hostapd.conf` file:

```
ssid=LXFwireless
wpa_passphrase=secret passphrase
interface=wlan0
auth_algs=3
channel=6
driver=nl80211
hw_mode=g
logger_stdout=-1
logger_stdout_level=2
```

```
max_num_sta=5
rsn_pairwise=CCMP
wpa=2
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP CCMP
```

The **driver** option might require some trial and error – most hardware will work with the **nl80211** driver, but you might need something else here. Test your *hostapd* configuration using the following:

```
$ sudo hostapd -d /etc/hostapd/hostapd.conf
```

If you don't see any errors, then try and connect to your hotspot with your phone. It probably won't get anywhere since there's no DHCP service at this stage, but you should at least be able to authenticate. We'll get DHCP for free once we set up our network bridge.

» `$ sudo service dnsmasq start`

Now set all the other internal network machines to use DHCP on the local network and you should discover two things: That the internal network machines can all ping each other (by IP address, once you've figured out who's who) and, further, that while they can't ping the outside world, they can at least perform name resolution. *Dnsmasq* will cache queries too, which saves a few milliseconds, should you query the same address more than once. When the client machines obtain a DHCP lease, *dnsmasq* will push the required gateway and DNS settings automatically, and the clients' `/etc/resolv.conf` files will be updated accordingly.

Packet forwarding

In order to get traffic flowing we need to set up packet forwarding and NAT (Network Address Translation). The first thing to do is to enable packet forwarding in the kernel. You can do this through the `/proc` interface like so:

```
$ echo 1 > /proc/sys/net/ipv4/ip_forward
```

To make this persist across reboots depends on your distro – some still use the `/etc/sysctl.conf` file, so add:

```
net.ipv4.ip_forward=1
```

Some (Arch and derivatives) have deprecated this file in favour of individual files in `/etc/sysctl.d/`. If this is you create a file, say `/etc/sysctl.d/40-ip-forward.conf`, and add the above line to it. That tells the kernel that forwarding is allowed, but we still have to stipulate the whences and wheretos. We'll use some good old fashioned *iptables* rules to do this. It's certainly possible with the newer *nftables* framework, but with every distro running a different version, and it still being quite new code, we won't do that. Since we're assuming our gateway machine is behind another gateway (that connects to your ISP), we needn't worry too much about security. So we can tell *iptables* to forward packets from the outside to our internal network:

```
$ sudo iptables -A FORWARD -i int0 -o ext0 -j ACCEPT
```

If you haven't previously added any *iptables* rules of your own then it's very likely you won't need the previous line –

most distros will, by default, allow all packets to be forwarded, as well as allowing everything else. We will need to set up NAT, so that traffic coming out of `ext0` can find its way back to `int0`. This step changes the packets' source addresses to our gateway's IP, and keeps track of the connection so that, as if by magic, any responses are forwarded to the correct machine on our internal network.

```
$ sudo iptables -t nat -A POSTROUTING -o ext0 -j MASQUERADE
```

A more robust approach here is to use SNAT (Source NAT) instead of MASQUERADE. This is conditional on the `ext0` interface on your gateway having a static IP, though. Note that this has nothing to do with how your ISP assigns IP addresses, merely how you set up your gateway machine's `ext0` interface. Using the following line instead means that NAT-ed connections can better survive link loss:

```
$ sudo iptables -t nat -A POSTROUTING -o ext0 -j SNAT --to-source 192.168.1.127
```

If `ext0` does not have the IP address above, then this won't work, funnily enough. The reason this doesn't work with a variable address is that if the link is interrupted then it's possible that upon reconnecting `ext0` is assigned a different address. The MASQUERADE target just forgets everything in the event of link loss, so connections can be safely (but slowly) re-established via the new address.

If you're running services on your internal network, and you want these to be accessible from the external network, then you must set up port forwarding, eg if you have an SSH server running on the machine **10.0.1.1**, in order for externally networked machines to see it, they'll need to connect via the gateway machine **192.168.1.254**. Since you might have an SSH server running on the gateway already, we'll forward its TCP port 2222 to port 22 on the internal machine:

```
$ sudo iptables -t nat -A PREROUTING -i ext0 -p tcp --dport 2222 -j DNAT --to 10.0.1.1:22
```

Now you can connect from the external network by pointing your SSH client at your gateway's port 2222. If you wanted to connect from the outside world, then you could add another rule which forwards traffic from your primary router to our gateway machine's port 2222. Once you've got everything working it's good to save your *iptables* rules. Each distro does this slightly differently, but ultimately rules are saved to a text file with:

```
$ sudo iptables-save > /etc/iptables.rules
```

Some distributions will auto-magically restore firewall rules from this file, others require to be told. Species of the Debian lineage, for example, would require the following line to be added to `/etc/network/interfaces`:

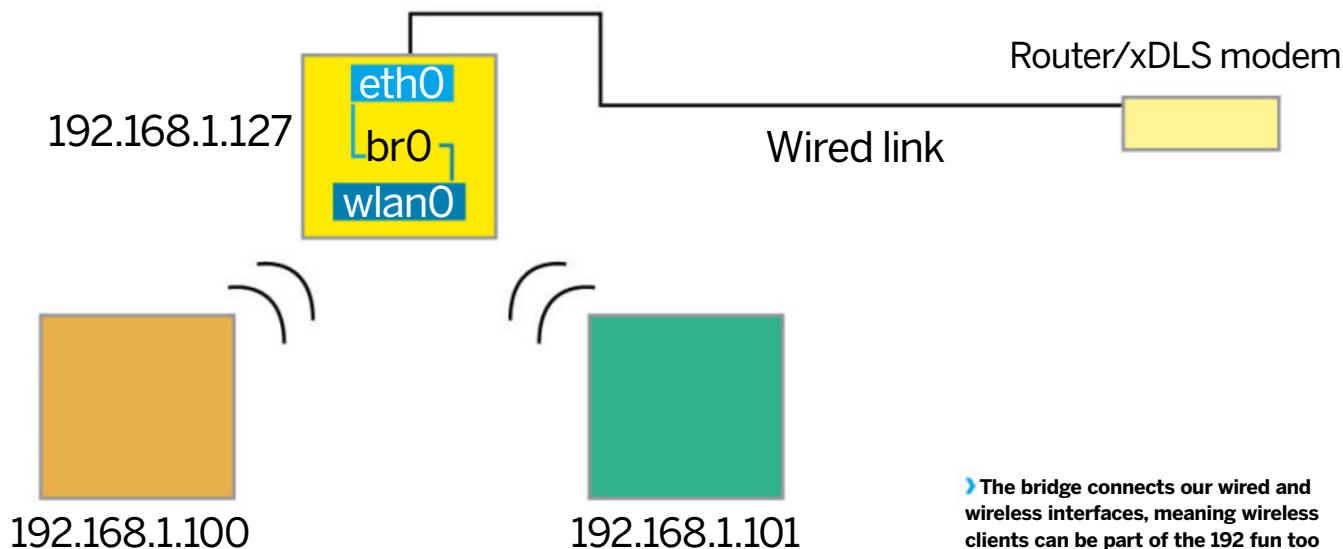
```
pre-up iptables-restore < /etc/iptables.rules
```

Cutting cords

Sometimes it's desirable to make your router accessible wirelessly, so that your mobile devices can connect to it. This is particularly useful if you find yourself using said devices in areas of your house with poor wireless coverage: Your gateway machine may be connected to your ISP-supplied box via Powerline Ethernet, and situated in or around the blackspot, bringing light to the darkness. It is also possible for this to work if there is a wireless link instead of a Powerline one, so that we have a simple wireless repeater. This may require two wireless cards in the gateway machine though, since some cards cannot be access points and clients

```
hostapd_new_assoc_sta: reschedule ap_handle_timer timeout for 14:30:c6:41:76:52
}
nl80211: Drv Event 19 (NL80211_CMD_NEW_STATION) received for wlp0s11flu4
nl80211: New station 14:30:c6:41:76:52
wlp0s11flu4: Event EAPOL_TX_STATUS (40) received
IEEE 802.1X: 14:30:c6:41:76:52 TX status - version=2 type=3 length=95 - ack=1
WPA: EAPOL-Key TX status for STA 14:30:c6:41:76:52 ack=1
WPA: Increase initial EAPOL-Key 1/4 timeout by 1000 ms because of acknowledged fr
wlp0s11flu4: Event EAPOL_RX (26) received
IEEE 802.1X: 121 bytes from 14:30:c6:41:76:52
IEEE 802.1X: version=1 type=3 length=117
WPA: Received EAPOL-Key from 14:30:c6:41:76:52 key_info=0x10a type=2 key_data=ler
WPA: Received Key Nonce - hexdump(len=32): 60 9e b0 35 de 70 06 6b ff 72 fb 82 7e
87 19 7e 19 1d 9f 8f a3 2d
WPA: Received Replay Counter - hexdump(len=8): 00 00 00 00 00 00 00 01
wlp0s11flu4: STA 14:30:c6:41:76:52 WPA: received EAPOL-Key frame (2/4 Pairwise)
WPA: 14:30:c6:41:76:52 WPA_PTK entering state PTKCALCNEGOTIATING
Searching a PSK for 14:30:c6:41:76:52 prev_psk=(nil)
WPA: PTK derivation - A1=00:13:ef:c7:00:16 A2=14:30:c6:41:76:52
WPA: Nonce1 - hexdump(len=32): 2c 5b 80 d7 fe 58 5b 42 43 ef 52 5b f7 a3 da a5 1c
70 00 de b5 18
WPA: Nonce2 - hexdump(len=32): 60 9e b0 35 de 70 06 6b ff 72 fb 82 7e 75 e6 b3 4c
1d 9f 8f a3 2d
WPA: PMK - hexdump(len=32): [REMOVED]
WPA: PTK - hexdump(len=48): [REMOVED]
WPA: 14:30:c6:41:76:52 WPA_PTK entering state PTKCALCNEGOTIATING2
WPA: 14:30:c6:41:76:52 WPA_PTK entering state PTKINITNEGOTIATING
wlp0s11flu4: STA 14:30:c6:41:76:52 WPA: sending 3/4 msg of 4-Way Handshake
WPA: Send EAPOL(version=2 secure=1 mic=1 ack=1 install=1 pairwise=1 kde_len=46 ke
Plaintext EAPOL-Key Key Data - hexdump(len=56): [REMOVED]
WPA: Use EAPOL-Key timeout of 100 ms (retry counter 1)
wlp0s11flu4: Event EAPOL_TX_STATUS (40) received
```

» This is what a successful WPA2 handshake should look like, but our hotspot fails to make it warm in here [see *Hotspots With Hostapd*, p129].



simultaneously. Furthermore, some cards lack Linux support for Access Point mode entirely. To check the capabilities of your wireless adapter use the **iw** tool (**sudo apt-get install iw**, if you don't have it). Running

```
$ iw list
```

will show you everything you could possibly want to know about your wireless devices. Check the 'Supported interface modes' section and ensure that 'AP' features somewhere. For a repeater set up you'll want to check 'valid interface combinations' for 'AP, mesh point'. Bear in mind that mesh protocols haven't yet been standardised so we won't try and cover them here. You'll need to be sure your wireless driver supports 4-address frames if you want to have a go.

Building bridges

So let's reverse our previous setup – we'll have an internet-facing wired connection to the gateway machine and we want to set up a local wireless network which provides access to this. In the previous section we used NAT to direct traffic between the internal and external networks, for this part we'll show you the alternative: network bridging. In many ways this is easier to understand than NAT, since it effectively unifies the two (or more) networks – providing a transparent bridge over which traffic flows unimpeded. However, the NAT approach is a little more flexible and provides more granular control, so you may prefer to use it again.

Whatever your preference, the first task is to install and configure the **hostapd** package (see the *Hotspots With Hostapd* box, p129).

A network bridge amalgamates two disparate interfaces (eg our **eth0** and **wlan0**) into a single interface, which in our example will be **br0**. Traffic will flow unimpeded between the two interfaces, and machines connected to our wireless hotspot will behave exactly as if they were connected to the same network as **eth0**. Hopefully, anyway. The first thing we'll need to **sudo apt-get install bridge-utils**, or equivalent. Then we create a new bridge with

```
$ sudo brctl addbr br0
```

and then add our wired interface to it with:

```
$ sudo brctl addif br0 eth0
```

We can't add the **wlan0** interface to the bridge without first starting our access point, since only devices in so-called promiscuous mode can be bridged. We can automate this by adding the following line to our **hostapd.conf**:

```
bridge=br0
```

Now we can test our access point by starting the service:

```
$ sudo service hostapd start
```

Replace with **systemctl start hostapd** if you're using Systemd. If everything works then enable the service. You can recreate the bridge setup in *Network Manager*, or by modifying **/etc/network/interfaces**.

And that concludes our foray into the world of bespoke routers. We've covered two different set ups, but don't be afraid to mix and match if you require, all these technologies will work together where it makes sense. We'll also be covering replacing the firmware on your home router with Linux one of these days. For now we're all huddled round an overclocked Raspberry Pi for Wi-Fi warmth.

Anonymising gateway

A nice thing about this setup is that any routing beyond the gateway machine propagates trivially to our internal network. So if you have your gateway machine's traffic routed through a VPN or *Tor*, then so can all our internal network traffic. Installing *Tor* is straightforward, the default config sets up a SOCKS proxy which you must connect your applications to. You can make this proxy available to the internal network by adding a line, such as **SOCKSProxy 10.10.254:9050** and **TransPort 10.10.254:9040** to **/etc/tor/torrc**.

Then you can set up any applications running on our internal network to use that proxy. A more passive approach is to set the gateway machine up as a transparent proxy or an isolating proxy, but this is beyond the scope of this little box.

You may subscribe to a commercial VPN provider, or even be running your own OpenVPN server somewhere on the internet. Either way, connection is usually achieved by setting up a TUN device (usually called **tun0**) on the client end. The routing table is then modified so that all

traffic travels via **tun0**, redirecting everything through the VPN. If you have already successfully set this up on your gateway machine, then all that is required is a small addition to our NAT rules:

```
$ sudo iptables -t nat -A POSTROUTING -o tun0 -j MASQUERADE
```

Since VPN traffic travels through the **tun0** interface, this line ensures that it can still find its way back to the internal network. As before, you can use SNAT here if **ext0** has a static IP.

Minix 3: Test a microkernel OS

Minix 3 has gone way beyond its educational origins with ARM compatibility, but microkernel reliability is still at its heart.



Minix 3 is a microkernel-based Unix clone, targeted towards robustness, reliability, and a small memory footprint. In a microkernel, drivers and servers are isolated, running in User Mode, and restricted in the effects they can have on the rest of the system (more on this later). The price for this is a small performance overhead, and some design challenges, but performance is not as important here as security and stability for most users.

Microkernels have a long history. In the '80s, academics were convinced they were the only way forward for OS development, and Minix author Andrew Tanenbaum never considered designing Minix in any other way. It also meant Richard Stallman turned to a microkernel as the basis of the Hurd, the kernel of the GNU OS. The imminent arrival of GNU – and BSD Unix – motivated Tanenbaum to keep Minix as an academic OS, but that fitted well with the aims of having something small enough to teach from, and cover in his famous book, *Operating Systems: Design and Implementation*.

Legal problems, and then technical challenges, held up the development of the Hurd, but Minix's tight focus enabled Tanenbaum to produce Minix 1 more or less single-handed, and three decades of putting the few thousands of lines of kernel code under close scrutiny while his doctoral students at Vrije Universiteit, Amsterdam worked on every area, have built a solid foundation for Minix 3.

Despite historical spats with Linux (see *Ancient history, opposite*) and the education-targeted early releases, Minix has become an appealing OS for many users, particularly as it's gradually integrated the NetBSD userland. This has been achieved largely because of millions in EU funding focused on 'secure and reliable' OS development, and many generous contributions from the Google Summer of Code.

The Minix team presented the latest version at FOSDEM this year and the 'MINi UnX' has grown in scope, but not in bloat. ARM support, for instance, has made it into the master code branch, and the team say that version 3 has "more complete source compatibility with NetBSD in terms of utilities, calls, types (lots of 64-bit), toolchain, codebase and packages", with "all Minix-specific code in a top-level **minix/** subdir".

Minix 3.3.0 isn't really production-ready, but it could make a stable base for a number of projects, so we'd encourage you to try it, whether on a BeagleBoard or a virtual machine, and keep an eye on developments.

Unfortunately, for those wanting to tinker outside of ARM boards on an old PC, in a desktop fashion, there's a major stumbling block: the X Window System, which was working in the previous release (based on a monolithic XFree86 server), has spectacularly broken. At time of writing, the current release of Xorg was being ported, but still wasn't ready, so it's command-line-only for now. You can, however, download previous releases if you really want to try the desktop. Not all NetBSD packages will run, but compatibility is improving and building from source may enable you to get favourite packages onto your system that aren't in the repos.

Inside the (virtual) box

Minix 3 will install on your x86 PC, but hardware compatibility is limited. Any i586 processor or later should be fine, although problems have been reported on Pentium 4, and the system can work with as little as 32MB of RAM. Peripherals are another matter, however: the list of compatible network cards is short, but includes those emulated by *Bochs*, *Qemu*, *VirtualBox*, *Virtual PC* and *VMware*, as well as the BeagleBone's LAN8701A. The remainder of the list is mostly old classics, which will seem familiar if you happen to have experimented with alternative OSes from a decade or more ago: 3Com 509, NE2000, Realtek RTL8139. At least the Intel PRO option is one you're more likely to have on hardware that hasn't been consigned to the attic.

It's probably better to grab a recent *Qemu* or *VirtualBox* from your distro's repository; we did most of our installs on

Ancient history: Tanenbaum vs Torvalds

The Minix 1 source code was available on floppy disks, and in the appendix to Andrew S. Tanenbaum's 1987 book *Operating Systems: Design and Implementation*. Soon a Usenet group grew around Minix with 40,000 subscribers, including Linus Torvalds who added new features. However, he grew frustrated with Tanenbaum's unwillingness to let Minix grow away from its tight educational focus, so in 1991, Torvalds announced on **comp.os.minix**: "I'm doing a (free) operating system (just a hobby,

won't be big and professional like GNU." When Torvalds was later accused in a book of stealing Minix code, Tanenbaum defended him, but rather waspishly said: "Linus didn't sit down in a vacuum and suddenly type in the Linux source code. He had my book, was running Minix, and undoubtedly knew the history (since it's in my book). But the code was his. The proof of this is that he messed the design up."

The Tanenbaum–Torvalds debate, revisited in the appendix to the 1999 book *Open Sources*:

Voices from the Open Source Revolution was a **comp.os.minix** thread over two weeks early in 1992, over the merits of monolithic kernels and microkernels – starting with a Tanenbaum contribution best summed up as 'monolithic kernels are obsolete.' It trailed off inconclusively, but after a few years many claimed 'Linus won', simply because of Linux's market share. The debate continues, but you can find Tanenbaum's more recent thoughts on the matter at <http://bit.ly/ReliableOS>.

the latter. Click the big New button at the top left of the *VirtualBox* window, and name your VM. We used **MINIX3**. Choose Other in both dropdown menus for OS type and version; on the next screen give it as much RAM as you can spare. We'd suggest that 256MB is adequate for Minix 3, but naturally more is better while you're exploring a new environment and pushing to see what it can do. We kept the default 2GB disk size on one VM install, and expanded to 8GB for another. Be inclined towards the latter if you're going to install *everything* Minix 3 has to offer, but bear in mind 3.3.0's lack of some software from previous releases.

Click Create and you'll see a MINIX3 VM listed in the left pane. Before starting, go to Settings, and tick the Hardware clock in UTC time checkbox, and check that Storage points to your downloaded ISO file. You can now start up the VM from the GUI. On a Core 2 Duo machine, without VT-x/AMD-V/ nested paging, we had to start *VirtualBox* with:

```
VBoxSDL --startvm MINIX3 --norawr0 --norawr3
```

Now follow the instructions in the walkthrough overleaf. Once the installation is complete, use **poweroff** instead of **shutdown** to halt the machine. Go back to Settings > Storage in *VirtualBox*, get rid of the ISO, and point at the newly created virtual disk image instead. Now boot with *VirtualBox*'s Start button (or use the workaround on PCs without VT-x and AMD-V virtualisation extensions).

Running on a BeagleBoard

With 3.3.0, the ARM port of Minix is finally integrated in the master code branch, along with the official x86 port. Targeted at the BeagleBoard, with its Cortex-A8-based system on a chip (SoC), it runs with varying degrees of success on the

BeagleBoard-xM (and its *Qemu*-based emulator), the BeagleBone, and the BeagleBone Black. Pre-built 3.3.0 images are available, but you'll miss out on the latest developments. To build a disk image for your BeagleBoard, make a directory at **~/minix** and **cd** to it – or to wherever you prefer to do the cross-compile – then get the source code:

```
$ git clone git://git.minix3.org/minix minixsrc
```

Now **cd** to **minixsrc/** and make a **.settings** file for BeagleBoard-xM use:

```
# beagleboard-xm
U_BOOT_BIN_DIR=build/omap3_beagle/
CONSOLE=tty02
```

And for the BeagleBone:

```
#beaglebone (and black)
U_BOOT_BIN_DIR=build/am335x_evm/
CONSOLE=tty00
```

Provided you have g++, the GNU C++ compiler, installed, the build tools in Minix will provide you with everything almost else. On Ubuntu you'll also need to:

```
apt-get install zlibc zlib1g zlib1g-dev
```

The build tools are based on NetBSD's **build.sh**. Call the ARM-specific version with:

```
./releasetools/arm_sdimage.sh
```

and you should find a **minix_arm_sd.img** waiting for you to copy across to your SD card, for booting your BeagleBoard or BeagleBone:

```
sudo dd if=minix_arm_sd.img of=/dev/mmcblk0 bs=1M
oflag=direct
```

Put the card in your BeagleBoard, boot up and you can log in as root. Ethernet isn't working on the BeagleBone, but on the BeagleBoards run **netconf** and select the LAN8710A. USB support on the BeagleBone is tagged as experimental, and there are known issues with hot-plugging on USB hubs in 3.3.0. Audio and the analogue-to-digital aren't yet working either, but GPIO is.

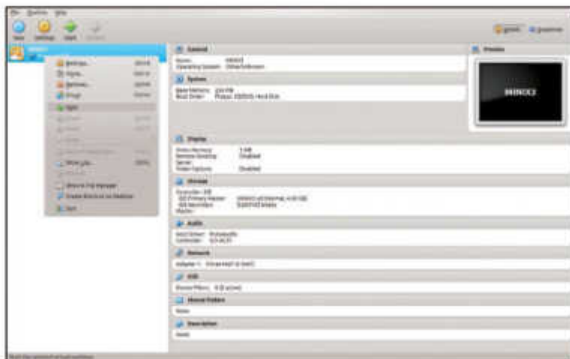
This is a promising first release of the port, and if you've got a BeagleBoard, it's a good sandbox for trying out Minix, particularly as this port will receive a lot of developer time, so you'll see improvements if you keep checking out the code. You can also try the ARM port under emulation with *Linaro Qemu*, the custom version of *Qemu* for the BeagleBoard-xM. You can find instructions at: <http://bit.ly/LinaroQemu>.

Reliability through architecture

Although at an early stage, the ARM port makes sense for an OS that sells itself on reliability and low resource use. The embedded ARM space covers thousands of different devices where the job of the operating system is, essentially, not to fall

Quick tip

So often *nix problems come down to either permissions or DNS. If you can't get pkgin to work after you install on *VirtualBox* with NAT, put our old friend **8.8.8.8** in **/etc/resolv.conf**, instead of **127.0.0.1**.



» **VirtualBox** is your best bet for trying out Minix 3, sweeping away hardware compatibility issues.

» over. In consumer devices the mean-time-to-failure should be longer than the life of the device, and Andrew Tanenbaum's stated aim of being finished when "computers don't need a reset button" seems even more applicable here.

Minix's reliability comes from modularity and fighting bloat. There are 6,000 lines of code in the kernel (see *A minimalist codebase*, foot of page). Outside of those 6 KLoC (which manage little more than interrupts, scheduling, and message passing) Minix runs everything else in the OS (all the drivers, from the console to the disk drive) in User Mode protecting against bugs and malicious attacks alike. To access memory, drivers and servers have to ask the kernel. No process is permitted to access anything beyond what it needs: the audio driver cannot access the disk drive, for example. The principle of least authority only allows kernel calls relevant for that class of driver, and allocates time slices to prevent infinite loops bringing down the system.

On top of that driver layer sits virtual memory management, the process manager, file systems and the remainder that traditional Unix regards as part of the kernel. These run in User Mode, with the MMU turned on.

User programs sit above this, but all three layers in User Mode are just – as the kernel is concerned – user processes. A crash in the code would bring down the system under Linux, but Minix 3 would merely need to restart the process. The data store server saves the state from crashed drivers, which is retrieved by the new one starting as a replacement.

Monitoring all of the servers, and restarting them after a crash, if appropriate, is the reincarnation server.

This is safe to do as most failures are caused by random timing errors and races. Tanenbaum's team have tested this, injecting millions of faults to overwrite 100 machine instructions in the running Ethernet driver binary. They injected 800,000 faults into each of the three different Ethernet drivers, causing 18,000 driver crashes – every time, the reincarnation server automatically replaced the driver.

Of course, you could start attacking the kernel yourself – a comprehensive test suite is included – but first get your system the way you like it. As noted in the walkthrough, **repositories.conf** needs editing to have the correct Minix FTP address. Uncomment the NetBSD repo while you're there. If you've never been on a server where *vi* is the only editing option, I'd recommend installing *bsdgames* on your laptop, and playing command-line Robots, until HJKL keystrokes are in muscle memory. Now run:

```
pkgin update
```

If there's a problem, particularly via NAT on *VirtualBox*, our best guess is DNS, so put:

```
nameserver 8.8.8.8
```

in **/etc/resolv.conf**. Now you can use **pkgin** search to find packages, for example:

```
pkgin install vim
```

to install. If you made a larger hard disk partition, and want to install everything at once, use:

```
pkgin_all
```

If you want to delve deeper, start with the wiki at <http://minix3.org>, which has a good tutorial on writing device drivers, for example, and plenty of other developer documentation. The user docs aren't too bad, but occasionally pages lag behind by a release version for a while. Naturally, help on updating the wiki would be appreciated by the project as much as more coders – it's actually at the top of the wish list on the wiki!

A bright future for microkernels?

That concludes this overview of Minix 3 – and of microkernels in general. We might be a long way from a usable Hurd, but microkernels have proved their worth in QnX-powered cars, Cisco routers and elsewhere. Just as a stable alternative for embedded ARM development, Minix 3 is well worth considering. Given its (nearly integrated) NetBSD userland, Minix makes the most usable microkernel OS for those familiar with GNU/Linux, and we confidently expect good things in the next release.

```
File Edit View Search Terminal Tabs Help
richard@luggable: ~
richard@luggable:~/Dropbox/work/code/c/ARM/MINIX$ git clone git://git.minix3.org/minix3 minix3
Cloning into 'minix3'...
remote: Counting objects: 129833, done.
remote: Compressing objects: 100% (53795/53795), done.
remote: Total 129833 (delta 74054), reused 128086 (delta 73000)
Receiving objects: 100% (129833/129833), 87.21 MB | 22.00 KiB/s, done.
Resolving deltas: 100% (74054/74054), done.
Checking connectivity... done.
Checking out files: 100% (44647/44647), done.
richard@luggable:~/Dropbox/work/code/c/ARM/MINIX$ cd ./minix3/
ls
BRANCH: master (521fa31)
$ ls
bin  common  doc  external  gnu  lib  LICENSE  Makefile.inc  releaseinfo  share  tests  usr  bin
build.sh  distinfo  etc  games  include  libexec  Makefile  nls  nls  sys  tools  usr  share
richard@luggable:~/Dropbox/work/code/c/ARM/MINIX/minix3$
BRANCH: master (521fa31)
$ nano -settings
richard@luggable:~/Dropbox/work/code/c/ARM/MINIX/minix3$
BRANCH: master (521fa31) / 1 uncommitted change
$ ./releaseinfo/wrap_sdimage.sh
Sourcing settings from .settings
CONTENT # beagleboard-ub
CONTENT U_BOOT_BIN_DIR=build/wrap2_beagle/
CONTENT CONSOLE=tty02
Cloning into './releaseinfo/u-boot'...
remote: Counting objects: 210000, done.
remote: Compressing objects: 100% (53795/53795), done.
Receiving objects: 50% (105000/210000), 40.00 MB | 30.00 KiB/s
```

» Building the SD card image for a BeagleBoard uses the NetBSD **build.sh** script and downloads most of the build tools needed on the fly.

A minimalist codebase

A few years ago, Linux passed 15 million lines of code (LoC). Minix 3 has 6,000 (6 KLoC) in the kernel, and about the same again for the drivers (that's traditional kernel code running in User Mode). With a thousand times more code in Linux, we could assume that it has a thousand times more bugs. But it's worse than that.

All code has bugs, typically five to ten bugs per KLoC, although FreeBSD comes out somewhat better with three per KLoC.

At FOSDEM, Tanenbaum cited a Stanford study

showing that Linux driver code had three to seven times more bugs than the rest of the kernel – simply because there's no fun in looking at messy driver code, rather than the important (and interesting) algorithms in the core kernel. In Linux 70% of the code is drivers. In Windows, 85% of the crashes are caused by drivers (which are mostly third-party). Running those buggy drivers in kernel space means vulnerabilities can bring down the whole system; in protected memory, isolated from other processes, the

reach of vulnerabilities is strictly limited. Take Netfilter, which replaced Linux 2.2's *ipchains* to control packet direction – offering options for packet filtering, network address translation and port translation. Netfilter sits inside the kernel, whereas Minix Netfilter sits in user space. If malicious executable code targets Netfilter again (such as the 2.6 'packet of death' kernel vulnerability) in Minix the User Mode process would be compromised, but in Linux the whole system would be owned.

Rescatux: System repairs

All those commands feeling a bit complex? Repair common system problems without resorting to the command line.



Linux live CDs are a wonderful invention, they let you try new distros, show Linux off to your unenlightened friends, and fix broken systems. There are live distros aimed specifically at repairing damaged systems, but they have a common drawback. They all require a certain amount of expertise and most of us don't break our systems often enough to gain that sort of experience. When your computer sits there showing you nothing but a glum message from the bootloader, your main priority is fixing it as quickly as possible, not spending time using a search

engine from a live CD to try and find the correct Grub incantation for your situation. I consider myself reasonably knowledgeable about bootloaders, but I still don't break them so often that I feel comfortable fixing them from the command line without at least a cursory RTFM to check my options.

Prep for live surgery

What we need is a live distro that is designed for fixing common problems without a great deal of background knowledge or research, and preferably one that doesn't require typing long commands where an error could make the situation worse. What we need is something like Rescatux.

Rescatux boots like a typical live CD to a lightweight LXDE desktop, but the window that opens on startup is the key difference. Called Rescapp, this is a one-stop centre for fixing various problems. Rather than simply list them, let's look at some of the problems that can arise when a computer system starts misbehaving at a low level, and how Rescatux can be used to fix them. This is not for dealing with minor user-level problems, a live CD such as Rescatux is usually brought out when things go seriously wrong.

Many system recovery operations require you to be booted from a live CD, either because normal booting is broken or because you need your root filesystem to be unmounted. You normally also need to use command line tools, and Rescatux provides all of this, but the Rescapp makes life much easier for many tasks.

When you press any of the operation buttons in Rescapp, it does not directly perform the operation. It displays a documentation page explaining how to use the option and, considering the low-level aspect of many of the operations, it's a good idea to read this. Then press the Run! button at the top right to perform the operation.

#1 Hard disk errors during boot

The first step when filesystem errors appear is to run **fsck** (that is short for filesystem check, not the expletive you use when you see the errors). This must be done while a filesystem is unmounted, hence the use of a live CD. Press the File System Check button. Rescapp temporarily mounts partitions in order to determine which distro they belong to. Of course, the corruption may prevent mounting (distro startup sequences are able to fix minor filesystem corruption transparently) so you may well be looking for one marked 'Cannot mount'. Only distro root directories can be identified,

› Rescatux works with 32 and 64 bit systems. Booting 32 on a 64 bit system is usually safe, but not optimal. The reverse will fail.

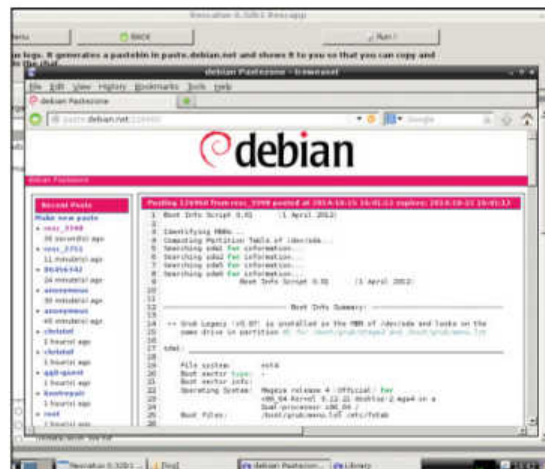


Getting more help

While Rescapp is able to fix many problems with a couple of mouse clicks, sometimes you need more help. Rescapp has some tools to help with this; first of all there is a web browser to search for answers, possibly using error messages you received when trying to repair your system. Everything Rescapp does is logged, the Show Log button presents you with a list of log files – remember, this is a live CD so you will only see logs from the current session. You can also view the logs directly, they are saved in **rescapp/logs**. Looking through the log for the operation you are attempting may give information useful to you. If it does not help you understand the problem, it may help others, which is where the Share logs button comes in. After selecting a log to share, Rescapp will send the log to a pastebin on Debian's servers and give you

the URL. Copy this somewhere safe and you can give it to anyone else for them to take a look at your logs. For a real time response, try the Chat button. This opens an IRC client to the **#rescatux** channel, where you can paste the URL of your pastebin and ask for help. You are not restricted to their own channel, of course, you could also try a channel dedicated to your distro for more specific advice. The 'Share log on forum' option works similarly, allowing you to ask for help on the **LXF** forums or your second favourite web forum.

Before you ask for help online, use the Boot Info Script button. This generates a file in **logs** containing information about your system, and you can share this with the 'Share log' option. Information about your system may be crucial to someone finding a solution to your problem.



▶ The Share Log button sends a log file to a pastebin and gives you the URL so you can share it with anyone who wants to help you.

if you have a separate **home**, it will appear as 'Not detected' (or 'Cannot mount' if it is damaged). There may be other reasons for a partition being unmountable; it may be your swap partition or an extended partition, so choose carefully. If in doubt, the Boot Info Script log (covered later) lists your partitions and their types.

#2 My password is not recognised!

Aside from boot merely resulting in an unfriendly grub> prompt, this is one of the most scary moments of computer use. You checked that you typed it correctly and that the caps-lock is not on. You may have forgotten it or, on a new install, have mis-typed it on setup.

Resetting a password involves booting a live CD and messing around with **chroots** – you cannot simply edit a file – or you can use Rescapp. Press the 'Change Gnu/Linux password' button and, after reading the explanation, press Run!, pick the distro (there will always be at least two, your installed distro and Rescatux, which appears as Debian 7) and then select the user to change. Enter the new password and the job is done. Try not to forget this one! This button is only for Linux passwords. If you run a dual-boot system with Windows, there is a separate option to reset your Windows password.

#3 I deleted the wrong files

It is both amazing and terrifying how a simple typing mistake can cause so much damage. For example if you meant to type

```
rm -f *.txt
```

but typed

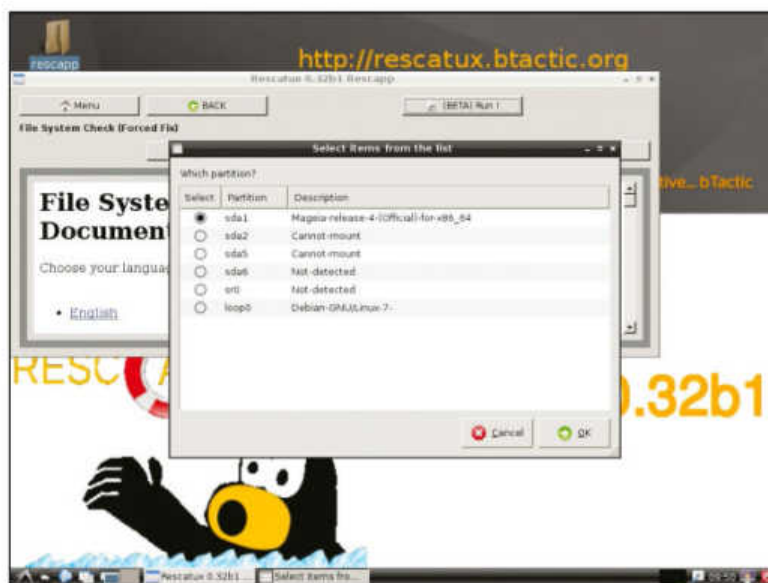
```
rm -f * .txt
```

instead. Or you wanted to reformat **/dev/sdc** but typed **sdb** instead. There is the odd filesystem-specific tool for recovering deleted files, such as **extundelete** – but a really determined typo can easily beat that, and it can't help if your partitions are gone. The first thing to do in such a situation is to stop writing to the disk – if the damage is on the partition containing your root filesystem you should shut down the

computer with

```
sudo shutdown -n
```

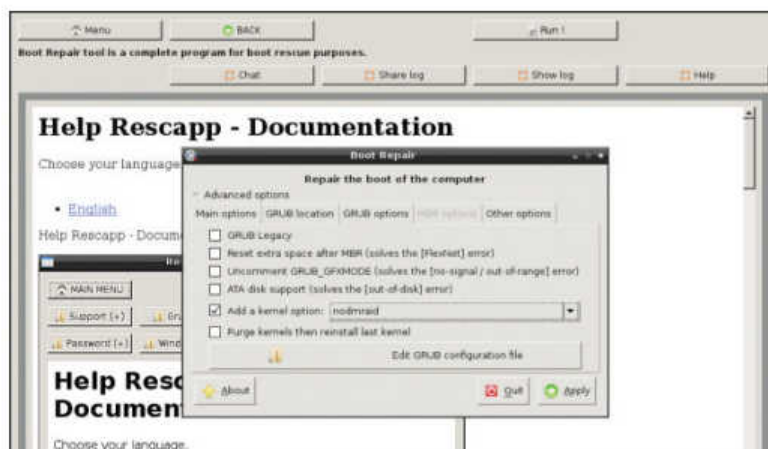
This kills processes without using the usual **init** system, which reduces the number of disk writes. Now you can boot Rescatux. If you partitioned a drive, you can use **testdisk** to search for the old partition boundaries and restore them. Repartitioning a drive only writes to the partition table, the actual data on the rest of the disk isn't touched until you format the new partitions. So if you can find the old partition boundaries and write them back into the partition table, everything should be back as it was. This is what **testdisk** does. After accepting the option to create a log file, that may be useful later, pick the disk to scan. The partition type should be Intel for the old-style MBR partition tables or EFI



▶ If a filesystem needs repair, select it and Rescapp will do the rest. Cannot-mount may mean damage, but here it indicates swap and extended partitions.

but you can also change the location of the bootloader and the options it uses. How often have you searched for a solution to an issue only to be told to “add option xyz to Grub”. You could go editing configuration files, but the Boot Repair window has a tab from which you can add various options without editing system critical files with the inherent risk of making things worse.

The current release of Rescatux is a beta and it does have some bugs. One or two of the options do not work properly, although we have not found anything dangerous. It's more a case of the option not doing anything rather than doing the wrong thing. We expect these bugs to be fixed in due course, but Rescatux is still worth keeping around as it can save a lot of heartache in many situations.



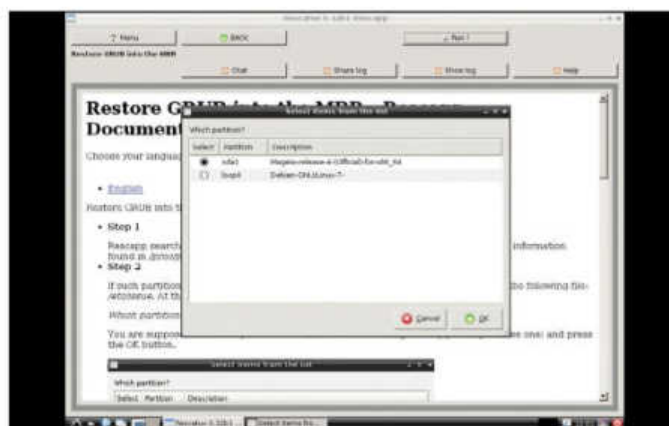
➤ Tweak your Grub options as well as performing other Grub backup and repair operation from the Expert tools section.

Fixing Grub



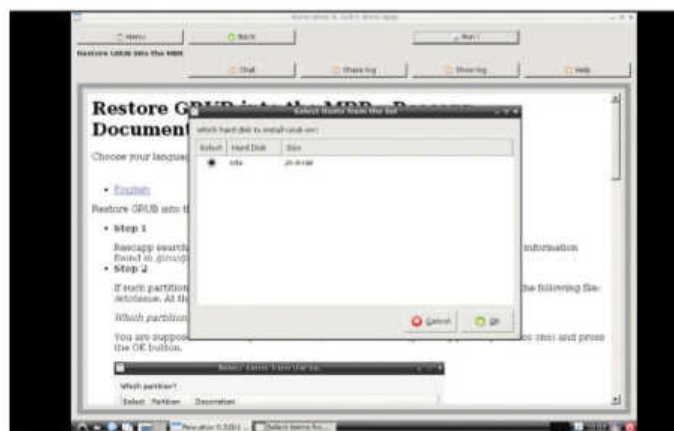
1 Restore Grub

Select Restore Grub from the main Rescapp window and read the help text to make sure you understand what is going on. Rescapp does most of the work, but you still need to know which distro and which disk you want to use for Grub. Press Run! when you are ready.



2 Select a distro

Rescapp will scan your disk partitions for installed distros, trying to recognise them by name when possible. Pick the one you want to use as your 'main' distro. This is the one that you are unlikely to want to remove, as that would involve repeating this process.



3 Grub selection

Now choose the disk to which you want to install Grub, usually sda. It doesn't have to be the one containing the distro you chose, but that is normally the case. If you boot from a USB stick, that may be recognised as sda with your hard disk on sdb.



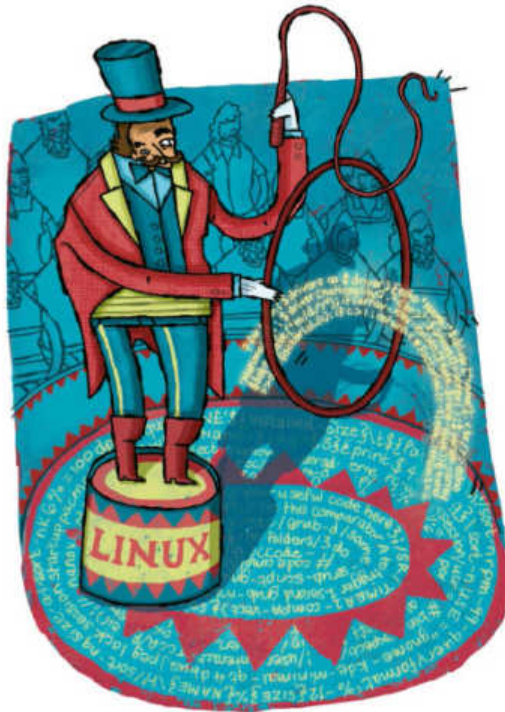
4 Auto-fix

Press OK, and Rescapp will perform the necessary steps, mounting and unmounting filesystems as needed and running **grub-install** with the correct **--boot-directory**, **--root-directory** and **device** arguments. Now you can reboot! It's as simple as that.

Wireshark:

Analyse traffic

We'll explain the necessary things that you need to know to start using Wireshark, and analyse three kinds of network traffic.



Wireshark is a very popular and extremely capable network protocol analyser that was developed by Gerald Combs. *Wireshark* was born in June 2006 when Combs renamed the network tool *Ethereal*, which he also created, as he was changing jobs and couldn't use the old name anymore. Nowadays, most people use *Wireshark*

and *Ethereal* has been consigned to history. Your Linux distribution will have a ready to install package for analyser too, so go ahead and install it.

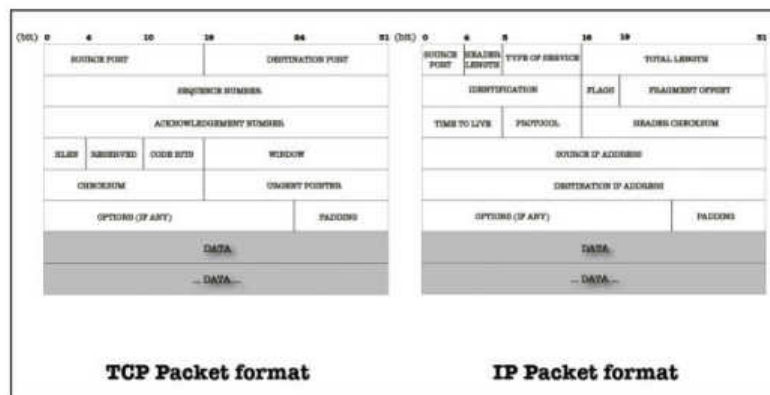
You may ask what makes *Wireshark* different to other network analysers – apart from the fact that it's free – and why we're not simply advocating using *tcpdump* for packet capturing? The main advantage of *Wireshark* is that it's a graphical application. Capturing and inspecting network traffic using a graphical user interface is a very helpful thing because it cuts through the complexity of network data.

To help the beginner understand *Wireshark* they will need to understand network traffic. The aim of this article then is to supply a comprehensive introduction to TCP/IP to enable you to come to useful conclusions about the network traffic data you're analysing.

If you run *Wireshark* as a normal user, you won't be able to use any network interfaces for capturing, because of the default Unix file permission that network interfaces have. It's more convenient to run *Wireshark* as root (**sudo wireshark**) when capturing data and as a normal user when analysing network data. Alternatively, you can capture network data using the *tcpdump* command line utility as root and analyse it using *Wireshark* afterwards. Please keep in mind that on a truly busy network, capturing using *Wireshark* might slow down a machine or, even worse, might not enable you to capture everything because *Wireshark* needs more system resources than a command line program. In such cases using *tcpdump* for capturing network traffic is the wisest solution.

Capturing network data

The easiest way to start capturing network packets is by selecting your preferred interface after starting *Wireshark* and then pressing Start. *Wireshark* will show network data on your screen depending on the traffic of your network. Note that you can select more than one interface. If you know nothing about TCP, IP or the other TCP/IP protocols, you may find the output complicated and difficult to read or understand. In order to stop the capturing process you just select Capture > Stop from the menu. Alternatively, you can press the fourth icon from the left, the one with a red square (which is shorthand for 'Stop the running live capture') on the Main toolbar (Note: its exact location depends on your *Wireshark* version). This button can only be pressed while you are capturing network data.



► The TCP packet and the IP packet format.

When using the described method for capturing, you can't change any of the default *Wireshark* Capture Options. You can see and change the Capture Options by selecting Capture > Options from the menu. There you can select the network Interface(s), see your IP address, apply capture filters, put your network card in promiscuous mode, and save your capture data in one or multiple files. You can even choose to stop packet capturing after a given number of network packets or a given amount of time or indeed a given size of data (in bytes).

Wireshark doesn't save the captured data by default but you can always save your data afterwards. It's considered good practice to first save and then examine the network packets unless there's a specific reason for not doing so.

Wireshark enables you to read and analyse already captured network data from a large amount of file formats including *tcpdump*, *libpcap*, Sun's *snoop*, HP's *nettl*, K12 text file etc. This means that you can read almost any format of captured network data with *Wireshark*. Similarly, *Wireshark* enables you to save your captured network data in a variety of formats. You can even use *Wireshark* to convert a file from a given format to another.

You can also export an existing file as a plain text file from the File menu. This option is mainly for manually processing network data or using it as input to another program.

There is an option that allows you to print your packets. I have never used this option in real life but it may be useful to print packets and their full contents for educational purposes.

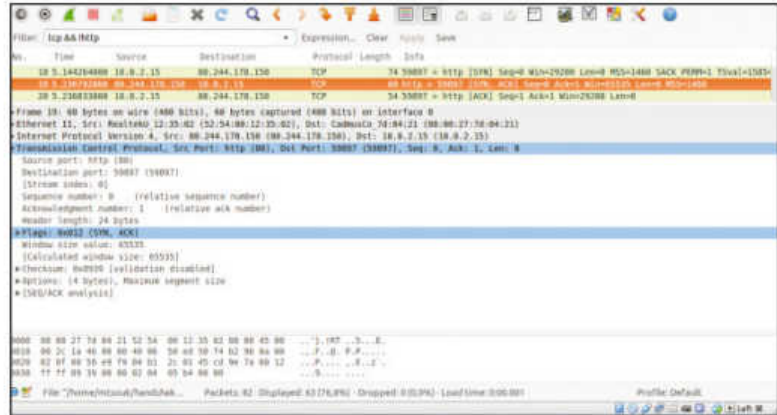
Display filters

While capture filters are applied during network data capture and make *Wireshark* discard network traffic that doesn't match the filter, display filters are applied after capture and 'hide' network traffic without deleting it. You can always disable a Display filter and get your hidden data back.

Generally, display filters are considered more useful and versatile than capture filters because it's unlikely you'll know in advance what you'll capture or want to examine. Nevertheless, applying filters at capture time can save you time and disk space and that's the main reason you might want to use them.

Wireshark will highlight when a display filter is syntactically correct with a light green background. When the syntax is erroneous, the background becomes pink.

Display filters support comparison and logical operators. The **http.response.code == 404 && ip.addr == 192.168.1.1** display filter shows the traffic that either comes from the 192.168.1.1 IP address or goes to the 192.168.1.1 IP address



that also has the 404 (Not Found) HTTP response code in it. The **!bootp && !ip && !arp** filter excludes BOOTP, IP and ARP traffic from the output. The **eth.addr == 01:23:45:67:89:ab && tcp.port == 25** filter displays the traffic from or to network device with the 01:23:45:67:89:ab MAC address that uses TCP port number 25 in its incoming or outgoing connections.

Keep in mind that display filters don't magically solve problems. They are extremely useful tools when used correctly but you still have to interpret the results, find the problem and think about the possible solutions yourself.

When defining rules please remember that the **(ip.addr != 192.168.1.5)** expression doesn't mean that none of the ip.addr fields can contain the 192.168.1.5 IP address. It actually means that one of the ip.addr fields should *not* contain the 192.168.1.5 IP address. Therefore, the other ip.addr field value can be equal to 192.168.1.5. You can think of it as 'there exists one ip.addr field that is not 192.168.1.5'. The correct way of expressing it is by typing **!(ip.addr == 192.168.1.5)**. This is a common misconception.

Also remember that MAC addresses are truly useful when you want to track a given machine on your LAN because the IP of a machine can change if it uses DHCP but its MAC address is more difficult to change.

It is advisable that you visit the display filters reference site for TCP related traffic at <http://bit.ly/WireSharkTCP>. For the list of all the available field names related to UDP traffic, it's advisable to look at <http://bit.ly/WireSharkUDP>.

About TCP/IP, TCP and IP

TCP/IP is the most widely used protocol for interconnecting computers and it is so closely related to the internet that it's

► The three packets (SYN, SYN+ACK and ACK) of a TCP 3-way handshake.

Quick tip

The fact that the FTP protocol usually uses port number 21 doesn't mean it's not allowed to use a different port number. In other words, don't blindly rely on the port number to characterise TCP/IP traffic.

The TCP protocol

TCP stands for Transmission Control Protocol. The main characteristic of TCP is that it's reliable and makes sure that a packet was delivered. If there's no proof of packet delivery, it resends the packet. TCP software transmits data between machines using segments (also called a TCP packet). TCP assigns a sequence number to each byte transmitted, and expects a positive acknowledgment (or ACK) from the receiving

TCP stack. If the ACK is not received within a timeout interval, the data is retransmitted as the original packet is considered undelivered. The receiving TCP stack uses the sequence numbers to rearrange the segments when they arrive out of order, and to eliminate duplicate segments.

The TCP header includes both the Source Port and Destination Port fields. These two fields, plus the source and destination IP addresses are

combined to uniquely identify each TCP connection. Ports help TCP/IP stacks in network connected devices (PCs and routers etc) to distribute traffic among multiple programs executing on a single device. If a service wants to be seen as reliable it's usually based on TCP, otherwise it's based on IP. But as you can imagine, reliability comes at a cost and therefore isn't always desirable.

385	1.191135000	64:70:02:ad:e9:44	b8:e8:56:34:a1:c8	ARP	60	10
386	1.193465000	d0:27:88:1d:d6:fb	b8:e8:56:34:a1:c8	ARP	60	10
387	1.194004000	d0:27:88:1d:d6:fb	b8:e8:56:34:a1:c8	ARP	60	10
388	1.196202000	00:1a:92:44:d7:67	b8:e8:56:34:a1:c8	ARP	60	10
389	1.210704000	b8:e8:56:34:a1:c8	Broadcast	ARP	42	Wh
390	1.210706000	b8:e8:56:34:a1:c8	Broadcast	ARP	42	Wh
391	1.210707000	b8:e8:56:34:a1:c8	Broadcast	ARP	42	Wh
392	1.210707000	b8:e8:56:34:a1:c8	Broadcast	ARP	42	Wh
393	1.210708000	b8:e8:56:34:a1:c8	Broadcast	ARP	42	Wh
394	1.210708000	b8:e8:56:34:a1:c8	Broadcast	ARP	42	Wh
395	1.210709000	b8:e8:56:34:a1:c8	Broadcast	ARP	42	Wh
396	1.222069000	b8:e8:56:34:a1:c8	Broadcast	ARP	42	Wh
397	1.222070000	b8:e8:56:34:a1:c8	Broadcast	ARP	42	Wh
398	1.222071000	b8:e8:56:34:a1:c8	Broadcast	ARP	42	Wh
399	1.222072000	b8:e8:56:34:a1:c8	Broadcast	ARP	42	Wh

Protocol size: 4						
Opcode: reply (2)						
Sender MAC address: d0:27:88:1d:d6:fb (d0:27:88:1d:d6:fb)						
Sender IP address: 10.67.93.21 (10.67.93.21)						
Target MAC address: b8:e8:56:34:a1:c8 (b8:e8:56:34:a1:c8)						
Target IP address: 10.67.93.11 (10.67.93.11)						

0000	b8	e8	56	34	a1	c8	d0	27	88	1d	d6	fb	00	00	00	01	..V4...
0010	00	00	06	04	00	02	d0	27	88	1d	d6	fb	0a	43	5d	15C].
0020	b8	e8	56	34	a1	c8	0a	43	5d	0b	00	00	00	00	00	00	..V4...C].....
0030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

► Part of an Nmap ping scan on a LAN as captured by Wireshark.

- » extremely difficult to discuss TCP/IP without talking about the Internet and vice versa. Every device that uses it has:
- **An IP address** This address must be unique at least to its local network.
 - **A network mask** Used for dividing big IP networks into smaller networks that's related to the current network.
 - **One or more DNS servers** Used for translating an IP address to a human-memorable format and vice versa
 - **A Default Gateway** This is optional if you want to communicate with devices beyond your local network. A Default Gateway is the network device that TCP/IP sends a network packet to when it doesn't 'know' where else to actually send it.

Every TCP service listens to a port that is unique to each machine. A machine that supports the HTTP protocol, the protocol that serves WWW, is also called an HTTP server. Similarly there exist FTP servers, DNS servers, etc. It's the two pairs of the IP addresses and port numbers on both ends of a TCP/IP interaction that uniquely identify a connection between two machines that use TCP/IP.

A TCP packet (*see the format of a TCP and an IP packet segment, pictured on p140*) can be used to establish connections; transfer data; send acknowledgements, advertise the buffer that holds incoming data, which is called Window Size, and close connections. As you can see in the packet screenshot (*see p141*), each TCP segment has a header part and a data part.

Quick tip

When you put your network card in promiscuous mode, you allow the network device to catch and read every network packet that arrives to it even if the receiver is another device on the network. Network packets still go to their original destination.

The TCP 3-way handshake

TCP provides a connection-oriented, reliable byte stream service. It's a full duplex protocol, which means that each TCP connection supports a pair of byte streams; one flowing in each direction. The term 'connection-oriented' means the two applications using TCP must first establish a TCP connection with each other before exchanging any data.

The TCP header includes a 6-bit flags field that's used to relay control information between TCP peers. The possible flags include SYN, FIN, RESET, PUSH, URG, and ACK. SYN and ACK flags are used for the initial TCP 3-way handshake as you will see in a while. The RESET flag signifies that the receiver wants to abort the connection.

The TCP three-way handshake goes like this: the client sends a TCP SYN packet to the server, and its TCP header includes a sequence number field that has an arbitrary value

in the SYN packet. The server sends back a TCP (SYN, ACK) packet which includes the sequence number of the opposite direction and an acknowledgement of the previous sequence number. Finally, in order to truly establish the TCP connection, the client sends a TCP ACK packet to acknowledge the sequence number of the server. After the TCP three-way handshake, the connection is established and is ready to send and receive data.

The traffic for this case was produced by running the following command:

```
$ wget http://www.linuxformat.com/
```

After some necessary DNS, ARP and ICMP network traffic, the TCP three-way handshake begins (*pictured top, p71*). The client IP address is 10.0.2.15 and the destination IP address is 80.244.178.150. A pretty simple display filter (**tcp && !http**) makes *Wireshark* display 63 out of 82 packets. The three packet numbers used in the handshake are sequential because the host wasn't performing any other network activity at the time of capturing, but this is rarely the case.

Ping scans

This part will examine the network traffic that's produced by *Nmap* when it performs a ping scan.

LAN ping scans are executed using the ARP protocol. Hosts outside a LAN are scanned using the ICMP protocol, so if you execute a *Nmap* ping scan outside of a LAN, the traffic will be different from one presented. In the example below, the *Nmap* command scans 255 IP addresses, from 10.67.93.1 to 10.67.93.255. The results show that at execution time only 10 hosts were up or, to be precise, only ten hosts answered the *Nmap* scan:

```
$ sudo nmap -sP 10.67.93.1-255
```

Starting Nmap 6.47 (<http://nmap.org>) at 2014-09-05 11:51 EEST

Nmap scan report for xxxxx.yyyyyy.zzzzz.gr (10.67.93.1)

Host is up (0.0030s latency).

MAC Address: 64:70:02:AD:E9:44 (Tp-link Technologies CO.)

Nmap scan report for srv-gym-ag-anarg.att.sch.gr (10.67.93.10)

Host is up (0.0051s latency).

MAC Address: 00:0C:F1:E8:1D:6E (Intel)

Nmap scan report for 10.67.93.20

Host is up (0.0066s latency).

MAC Address: D0:27:88:1D:15:20 (Hon Hai Precision Ind. Co.Ltd)

Nmap scan report for 10.67.93.21

Host is up (0.0053s latency).

MAC Address: D0:27:88:1D:D6:FB (Hon Hai Precision Ind. Co.Ltd)

Nmap scan report for 10.67.93.22

Host is up (0.0080s latency).

MAC Address: 00:1A:92:44:D7:67 (Asustek Computer)

Nmap scan report for 10.67.93.29

Host is up (0.057s latency).

MAC Address: 00:78:E2:47:49:E5 (Unknown)

Nmap scan report for 10.67.93.78

Host is up (0.0023s latency).

MAC Address: 00:80:48:24:6A:CC (Compex Incorporated)

Nmap scan report for 10.67.93.147

Host is up (0.028s latency).

MAC Address: 00:14:38:64:5D:35 (Hewlett-Packard)

Nmap scan report for 10.67.93.172

Host is up (0.016s latency).

```
MAC Address: 00:50:27:00:E4:F0 (Genicom)
Nmap scan report for www.yyyyyy.zzzzz.gr (10.67.93.11)
Host is up.
Nmap done: 255 IP addresses (10 hosts up) scanned in 1.25
seconds
```

The purpose of the ping test is simply to find out if an IP is up or not – see the grab on the opposite page. What's important for *Nmap* in a ping scan is not the actual data of the received packets but, put relatively simply, the existence of a reply packet. As all traffic is in a LAN, each network device uses its MAC address in the reply so you only see MAC addresses in both Source and Destination fields. The presence of a reply makes *Nmap* understand that a host is up and running. As a MAC address includes information about the manufacturer of the network device, *Nmap* also reports that information for you.

Nmap also calculates the round trip time delay (or latency). This gives a pretty accurate estimate of the time needed for the initial packet (sent by *Nmap*) to go to a target device, plus the time that the response packet took to return to *Nmap*. A big latency time is not a good thing and should certainly be examined.

Analysing DNS traffic

DNS queries are very common in TCP/IP networks. A DNS query creates little traffic and therefore it is an appropriate example for learning purposes. The following command will be used for generating the necessary DNS network traffic that will be examined:

```
$ host -t ns linuxformat.com
linuxformat.com name server ns0.future.net.uk.
linuxformat.com name server ns1.future.net.uk.
```

Two packets are needed in total: one for sending and one for answering the DNS query (see *grab, right*).

The first packet is number 3 and the second is number 4. A Display filter (DNS) is used to minimise the displayed data and reveal the useful information. The UDP (User Datagram Protocol) protocol was used and the desired information was sent back without any errors as shown by the Flags information. You can also tell by noting the time difference between the DNS query (1.246055000) and its answer (1.255059000) that the DNS services work fine because of the reasonable response time. The DNS server asked has the 10.67.93.1 IP address – as you can see from the destination IP address of the first packet. The same DNS server answered the DNS query as you can see from the source IP address of the second packet. The 'Answer RRs: 2' line informs us that there will be two answers for the DNS query. In time, you will

be able to take all this in with one glance.

UDP uses the underlying IP protocol to transport a message from one machine to another, and provides the same unreliable, connectionless packet delivery as IP. It doesn't use acknowledgements to make sure messages arrive, it doesn't order incoming messages, and it doesn't provide feedback to control the rate at which information flows between the machines. Thus, UDP messages can be lost, duplicated, or arrive out of order. Furthermore, packets can arrive faster than the recipient can process them.

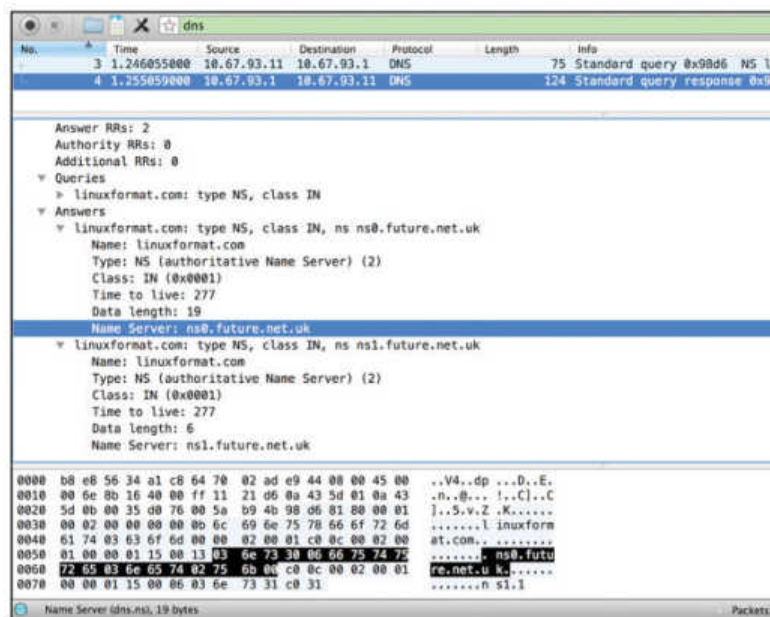
The destination port of the first packet is 53 which is the usual port number of the DNS service. The UDP part of the second packet shows the port numbers used for the reply:

```
User Datagram Protocol, Src Port: 53 (53), Dst Port: 53366 (53366)
Source Port: 53 (53)
Destination Port: 53366 (53366)
Length: 90
Checksum: 0xb94b [validation disabled]
[Stream index: 0]
```

As it happens with most tools, the more you use *Wireshark*, the more efficient you will become with it, so keep on practicing and learning!

Quick tip

There is also a console version of *Wireshark* called *tshark*. The two main advantages of *tshark* are that it can be used in scripts and that it can be used through an SSH connection. Its main disadvantage is that it does not have a GUI. *Tshark* can also entirely replace *tcpdump*.



► Here is how *Wireshark* shows the traffic of a DNS query after applying a Display filter. Notice the green colour around DNS that shows the validity of it.

The IP protocol

IP stands for Internet Protocol. The main characteristic of IP is that it's not a reliable protocol by design. Unreliable means that packets may not reach its destination for various reasons, including transmission errors, network hardware failures and network congestion. Networks may also deliver packets out of order, deliver them after a substantial delay or deliver duplicates. Nevertheless, a programmer can program reliable applications that use IP by implementing their own error-checking code but this is a non-trivial task.

When the information doesn't need many network packets, using a protocol that's based on IP is more efficient than using TCP, even if you have to re-transmit a network packet, because there's no three-way handshake traffic overhead.

IP encapsulates the data that travels in a TCP/IP network, because it's responsible for delivering packets from the source host to the destination host according to the IP addresses. IP has to find an addressing method to effectively send the packet to its destination. Dedicated devices that you'd recognise as

routers mainly perform IP routing but every TCP/IP device has to do basic routing.

Each IP address is a sequence of four 8-bit numbers, separated by dots. Each number has a value between 0 (=2⁰-1) and 255 (=2⁸-1). Example IP addresses are 10.25.128.254, 213.201.43.23 and 192.168.1.200.

IPv6 was developed by IETF and its purpose is to solve the problem of running out of IPv4 addresses. IP uses 32-bit addresses whereas IPv6 uses 128-bit addresses, offering more than 7.9×1,028 times as many as IPv4.

Subscribe to



Choose your LINUX package

Print £31.50 For 6 months

Every issue comes with a 4GB DVD packed full of the hottest distros, apps, games and a lot more.



Digital £22.50 For 6 months

The cheapest way to get *Linux Format*. Instant access on your iPad, iPhone and Android device.



Bundle £38.50 For 6 months

- » Includes a DVD packed with the best new distros.
- » Exclusive access to the *Linux Format* subscribers-only area – with 1,000s of DRM-free tutorials, features and reviews.
- » Every new issue in print and on your iOS or Android device.
- » Never miss an issue.



Get all the best in FOSS

Every issue packed with features, tutorials and a dedicated Pi section.



Subscribe online today...

myfavouritemagazines.co.uk/LINsubs

Prices and Savings quoted are compared to buying full-priced UK print and digital issues. You will receive 13 issues in a year. If you are dissatisfied in any way you can write to us or call us to cancel your subscription at any time and we will refund you for all undelivered issues. Prices correct at point of print. For full terms and conditions please visit: <http://myfavm.ag/magterms>. Offer ends 31/10/2015

LINUX MADE SIMPLE

Enjoy a hassle-free transition to Linux with our expert advice

Learn how to upgrade, customise and master the free OS with a collection of tutorials and guides from the Linux experts



148
pages of advice
from the makers of
**LINUX
FORMAT**

Dozens of expert tutorials

- Try out Ubuntu, Mint and more!
- Build a custom Linux PC or Steam box
- Learn the secrets of the command line



Discover the most amazing Linux apps for absolutely any task

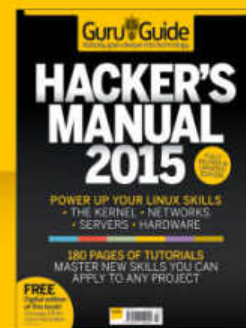
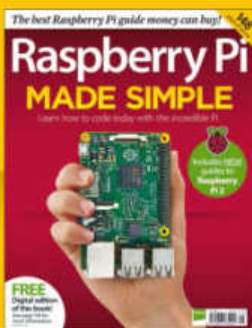


Try a minimal installation, make a rescue disk, and much more



Resurrect those old machines with low-powered Linux distros

LIKE THIS? THEN YOU'LL ALSO LOVE...



Visit myfavouritemagazines.co.uk today!